UNIVERSIDADE DE COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

FACULDADE DE CIÊNCIAS E TECNOLOGIAS

Projecto IC^3:

Uma plataforma Integrada de Computação e Comunicações

Tiago José dos Santos Martins da Cruz

COIMBRA

2005

Universidade de Coimbra

Departamento de Engenharia Informática

Faculdade de Ciências e Tecnologias

Projecto IC^3:
Uma plataforma Integrada de Computação e Comunicações

Tiago José dos Santos Martins da Cruz

Dissertação submetida para satisfação dos requisitos do programa
de Mestrado em Engenharia Informática

Coimbra

2005

Tese realizada sob a orientação do

Prof. Doutor Paulo Alexandre Ferreira Simões

Professor Auxiliar do Departamento de Engenharia Informática da

Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Palavras Chave

Gestão de Desktops

Sistemas Distribuídos

Integração computador-serviços de telefonia

Convergência de plataformas

Keywords

Desktop Management

Distributed Systems

Computer-Telephony Integration

Platform Convergence

Sumário

No momento em que o paradigma da computação pessoal concretizou a transição dos ambientes domésticos para o mundo empresarial, abriu-se um leque de perspectivas e possibilidades que mudou de forma radical o modo como os utilizadores encaram os meios informáticos. Esta mudança, aliada à difusão das redes de área local potenciou o surgimento de novas formas e processos de trabalho colaborativo que trouxeram um novo fôlego às organizações.

Como consequência desta evolução, deu-se um aumento do número de postos de trabalho informatizados ("*desktops*"), decorrente da progressiva democratização do PC (*Personal Computer*) e dos sistemas de informação, implicando uma necessidade cada vez mais premente de mecanismos de gestão eficazes do parque de PCs em uso.
Esta demanda é frequentemente relegada para um plano inferior no estudo da temática da gestão de redes e sistemas distribuídos, nem sempre sendo alvo do merecido reconhecimento. No entanto, constitui uma tarefa exigente, desgastante, consumidora de consideráveis quantidades de tempo e recursos, e com assinalável impacto no funcionamento das organizações. É contrariando esta tendência, e reconhecendo assim implicitamente a relevância do tema, que no Laboratório de Comunicações e Telemática (LCT) do Centro de Informática e Sistemas da Universidade de Coimbra se tem vindo a desenvolver trabalho de investigação devotado ao assunto, procurando compreender melhor a natureza dos problemas associados à gestão de *desktops*.

O trabalho subjacente a esta dissertação complementa esse trabalho anterior, focado essencialmente no desenvolvimento de mecanismos uniformes de gestão distribuída orientados para uso em parques informáticos heterogéneos. O presente trabalho privilegia, por contraste, questões relacionadas com a ineficiência e inadequação dos *desktops* correntes.

Nesse sentido, é proposta, desenvolvida e testada uma nova plataforma computacional, que conjuga alguns dos aspectos mais interessantes do PC clássico e de paradigmas alternativos como o *thin client* e o *network computer*, sem sacrificar a liberdade dos utilizadores.

Abstract

The transition process that moved the personal computing paradigm from the domestic environment to the enterprise unveiled a new set of perspectives and possibilities that radically changed the way users perceive and understand the purpose of the computational resources put at their disposal. These changes, associated with the spreading of local-area networks, laid the groundwork where new processes and models of collaborative work would emerge and evolve.

As a consequence of this paradigm shift, in parallel with the democratization of the PC *(Personal Computer)* and information systems, there was a dramatic increase in the number of computer-equipped workplaces, fueling demand for adequate desktop management mechanisms.

Desktop management is usually not recognized as a major research topic in the general area of distributed network and systems management. Nevertheless, desktop management is indeed a challenging task that absorbs vast amounts of time, financial funds and human resources. Going against this trend, and implicitly recognizing the relevance of the subject, the Laboratory of Communications and Telematics (LCT) of the Centre for Informatics and Systems of the University of Coimbra has long been involved in research work devoted to better understanding the nature of desktop management problems.

The work associated with this thesis thus complements previously research, focused on the search for uniform distributed management mechanisms appropriate for heterogeneous desktop management environments. In contrast, present work focuses directly on the efficiency and adequacy of current desktop platforms.

In this sense, a new computational platform is proposed, developed and tested. This platform is based on a hybrid concept – halfway between PCs and thin-clients/network-computers – that inherits the best features of PCs and thin-clients without sacrificing user freedom.

Agradecimentos

Ao meu orientador científico, Doutor Paulo Alexandre Ferreira Simões pelo incentivo, rigor, atitude construtiva e pela disponibilidade sempre demonstrada ao longo do desenvolvimento deste projecto.

Ao José Martins, ao Eng. Luís Pinto e ao Mestre Jorge Tavares pelos conselhos, amizade e apoio dispensados.

Ao Dr. José Marques, Director Executivo da AIRC pela sua atitude de tolerância, espírito humano e compreensão que me permitiu prosseguir este projecto em simultâneo com o desenvolvimento de uma actividade profissional.

À Olga pela infinita paciência, imenso carinho e precioso auxílio.

Aos meus pais pelo apoio e disponibilidade.

Ao melhor Amigo do mundo, pelo amparo e apoio nas horas difíceis.

Índice

Lista de figuras

Lista de Tabelas

Lista de acrónimos e abreviaturas

AAA	*Authentication, Authorization and Accounting*
ACK	*Acknowledge*
ACPI	*Advanced Configuration and Power Interface*
AES	*Advanced Encryption Standard*
AFS	*Andrew File System*
AGP	*Accelerated Graphics Port*
ALC	*Asynchronous Layered Coding*
ALSA	*Advanced Linux Sound Architecture*
ANSI	*American National Standards Institute*
AP	*Access Point*
API	*Application Programming Interface*
ARQ	*Automatic Repeat reQuest*
ASF	*Alert Standard Format*
ASIC	*Application Specific Integrated Circuit*
ASM	*Any Source Multicast*
ATA	*AT Attachment*
AVSG	*Available VSG*
AWG	*American Wire Grade*
BER	*Bit Error Rate*
BIOS	*Basic Input-Output System*
BITBLT	*Bit Block Transfer*
BSS	*Basic Service Set*
CBQ	*Class Based Queuing*
CF	*Compact Flash*
CFDP	*Coherent File Distribution Protocol*
CHAP	*Challenge Handshake Authentication Protocol*
CIFS	*Common Internet File System*
CIM	*Common Information Model*
CIR	*Committed Information Rate*
COM	*Component Object Model*
CoS	*Class of Service*
COTS	*Commercial Off The Shelf*
CPU	*Central Processing Unit*
CRC	*Cyclic Redundancy Check*
CRM	*Customer Relationship Management*
CRT	*Cathode Ray Tube*
CSMA	*Carrier Sense Multiple Access*
CSMA/CA	*CSMA / Collision Avoidance*
CSMA/CD	*CSMA / Collision Detect*
CVV	*Coda Version Vector*
DC	*Direct Current*
DCOM	*Distributed COM*
DDC	*Display Data Channel, Digital Data Channel*
DDR	*Double Data Rate*
DEI	Departamento de Engenharia Informática
DES	*Data Encryption Standard*
DHCP	*Dynamic Host Configuration Protocol*
DIMM	*Dual Inline Memory Module*
DIY	*Do It Yourself*

DMA	*Direct Memory Access*
DMI	*Desktop Management Interface*
DMTF	*Distributed Management Task Force*
DNS	*Domain Name Service*
DOMUS	*Domestic Oriented Multiservice Self-Managed Appliance*
DRI	*Direct Rendering Interface*
DRM	*Direct Rendering Manager, Digital Rights Management*
DSCP	*DiffServ Code Point*
DTD	*Document Type Definition*
DTE	*Data Terminal Equipment*
DTP	*Desktop Publishing*
EAP	*Extensible Authentication Protocol*
EAPOL	*EAP Over LAN*
ECB	*Electronic CodeBook*
EFI	*Extensible Firmware Interface*
EIA	*Electronic Industries Alliance*
EOL	*End Of Life*
ESS	*Extended Service Set*
FCTUC	Faculdade de Ciências e Tecnologia da Universidade de Coimbra
FEC	*Forward Error Correct*
FFS	*Fast File System*
FIFO	*First In, First Out*
FLUTE	*File Delivery over Unidirectional Transport*
FSB	*Front Side Bus*
GbE	*Gigabit Ethernet*
GNU	*GNUs Not Unix*
GPL	*GNU Public License*
GSM	*Global System for Mobile Communications*
GUI	*Graphical User Interface*
HAL	*Hardware Abstraction Layer*
HD	*Hard Disk*
HTB	*Hierarquical Token Bucket*
HTTP	*HyperText Transport Protocol*
I2C	*Inter Integrated Circuit*
iAMT	*Intel Active Management Technology*
IAX	*Intra Asterisk eXchange*
IC^3	*Integrated Communications and Computing Concept*
ICMP	*Internet Control Message Protocol*
IDE	*Integrated Drive Electronics*
IEC	*International Electrotechnical Commission*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IMAP	*Internet Message Access Protocol*
INRIA	*Institut National de Recherche en Informatique*
IPv4	*IP version 4*
IPv6	*IP version 6*
ISA	*Industry Standard Architecture*
ISO	*International Standards Organization*
JVM	*Java Virtual Machine*
LAN	*Local Area Network*
LCD	*Liquid Crystal Display*
LCT	Laboratório de Comunicações e Telemática
LDAP	*Lightweight Directory Access Protocol*
LEAP	*Lightweight EAP*

LSI	*Large Scale Integration*
LTSP	*Linux Terminal Server Project*
LVS	*Linux Virtual Server*
MAC	*Medium Access Control*
MAPI	*Mail API*
MCL	*MultiCast Library*
MDI	*Media Dependent Interface*
MG/MS	*Magic Gate / Memory Stick*
MIB	*Management Information Base*
MIPS	*Million Instructions Per Second*
MMC	*MultiMedia Card*
MOS	*Mean Opinion Scale*
MPLS	*MultiProtocol Label Switching*
MSCHAP	*Microsoft CHAP*
MSI	*Medium Scale Integration*
MTBCF	*Mean Time Between Critical Failures*
MTBF	*Mean Time Between Failures*
MTFTP	*Multicast Trivial File Transfer Protocol*
MTTR	*Mean Time To Repair*
NACK	*Non-Acknowledge*
NAS	*Network Attached Storage*
NBD	*Network Block Device*
NBP	*Network Boot Programs*
NDA	*Non Disclosure Agreement*
NORM	*Negative Acknowledgement Oriented Reliable Multicas*
OEM	*Original Equipment Manufacturer*
OOB	*Out Of Band*
OpenDMS	*Open Desktop Management Solution*
OSI	*Open Systems Interconnection*
OSS	*Open Sound System*
PAE	*Port Authentication Entity*
PAL	*Programmable Array Logic*
PAP	*Password Authentication Protocol*
PBX	*Private Branch eXchange*
PC	*Personal Computer*
PCI	*Peripheral Component Interconnect*
PCMCIA	*Personal Computer Memory Card International Association*
PD	*Powered Device*
PDA	*Personal Digital Assistant*
PEAP	*Protected EAP*
PHB	*Per-Hop Behavior*
PIM	*Personal Information Manager*
PLC	*Packet Loss Concealment*
PnP	*Plug and Play*
PoE	*Power over Ethernet*
PoEP	*Power over Ethernet Plus*
POP	*Post Office Protocol*
POS	*Point Of Sale*
POST	*Power On Self Test*
PPTP	*Point-to-Point Tunneling Protocol*
PSE	*Power Sourcing Equipment*
PXE	*Preboot eXecution Environment*
QoS	*Quality of Service*
RADIUS	*Remote Authentication Dial-In User Service*

RAM	*Random Access Memory*
RDIS	Rede Digital com Integração de Serviços
RDP	*Remote Desktop Protocol*
RF	Radio-Frequência
RFC	*Request For Comments*
RIS	*Remote Installation Services*
RISC	*Reduced Instruction Set Computer*
ROI	*Return Of Investment*
ROM	*Read-Only Memory*
RTP	*Real Time Protocol*
RTT	*Round Trip Time*
RVM	*Recoverable Virtual Memory*
SAN	*Storage Area Network*
SCSI	*Small Computer System Interface*
ScTP	*Screened Twisted Pair*
SD	*Secure Digital*
SDRAM	*Synchronous Dynamic RAM*
SFTP	*Secure FTP*
SIMM	*Single Inline Memory Module*
SIP	*Session Initiation Protocol*
SIPP	*Stable Image Platform Program*
SLA	*Service Level Agreement*
SMART	*System Monitoring, Analysis and Reporting Technology*
SMB	*Server Message Block*
SMBIOS	*Systems Management BIOS*
SNMP	*Simple Network Management Protocol*
SOHO	*Small Office, Home Office*
SoIP	*Services over IP*
SPOF	*Single Point of Failure*
SQL	*Structured Query Language*
SS7	*Signaling System 7*
SSH	*Secure Shell*
SSL	*Secure Socket Layer*
SSM	*Source Specific Multicast*
SUS	*Software Update Services*
SWOT	*Strengths, Weaknesses, Opportunities and Threats*
TCO	*Total Cost of Ownership*
TCP/IP	*Transmission Control Protocol / Internet Protocol*
TDM	*Time Division Multiplexing*
TFT	*Thin-Film Transistor*
TFTP	*Trivial File Transfer Protocol*
TI	Tecnologias da Informação
TIA	*Telecommunications Industry Association*
TLS	*Transport Layer Security*
ToS	*Type of Service*
TTL	*Transistor to Transistor Logic*
TTLS	*Tunneled TLS*
UDMA	*Ultra DMA*
UDP	*User Datagram Protocol*
UEFI	*Universal Extensible Firmware Interface*
ULSI	*Ultra Large Scale Integration*
UMA	*Unified Memory Architecture*
UNDI	*Universal Network Device Interface*
USB	*Universal Serial Bus*

USD	*US Dollars*
UTP	*Unshielded Twisted Pair*
VGA	*Video Graphics Array*
VLAN	*Virtual LAN*
VLSI	*Very Large Scale Integration*
VNC	*Virtual Network Computing*
VoIP	*Voice over IP*
VPN	*Virtual Private Network*
VRAM	*Video RAM*
VSG	*Volume Service Group*
WAN	*Wide Area Network*
WBEM	*Web-Based Enterprise Management*
WDS	*Windows Deployment Services*
WEP	*Wired Equivalent Privacy*
WfM	*Wired for Management*
WIMP	*Windows Icons Mouse and Pointer*
WLAN, WiLAN, Wi-Fi	*Wireless LAN*
WMI	*Windows Management Instrumentation*
WOL	*Wake On LAN*
WPA	*Wi-Fi Protected Access*
WQL	*WMI Query Language*
WTS	*Windows Terminal Services/Server*
WUS	*Windows Update Services*
XML	*eXtensible Markup Language*
ZAW	*Zero Administration for Windows*

1. Introdução

Neste capítulo é discutida a motivação subjacente a este trabalho, enquadrada na temática específica da gestão de *desktops* e do conjunto de questões-chave que constituiram o ponto de partida para o trabalho desenvolvido. Pretende-se, de igual modo, identificar os objectivos definidos à partida.

A estrutura desta dissertação é também apresentada neste Capítulo,.

1.1 Motivação

O aparecimento do computador pessoal e a afirmação da gestão de *desktops* – enquanto área de interesse da investigação em redes e sistemas distribuídos – constituem um par causa--efeito indissociável cujo relacionamento se baseia num ciclo de influência mútua. A gestão de *desktops* é cada vez mais complexa e relevante, mas esse agravamento nem sempre tem sido correspondido pelo desenvolvimento de recursos e soluções à altura.

A problemática da gestão de *desktops* nem sempre tem sido justamente reconhecida, apesar da sua considerável relevância no actual contexto das TI (Tecnologias da Informação). Ainda que seja porventura considerada como desinteressante – devido a uma aparente falta de especialização ou complexidade tecnológica – esta actividade acarreta consigo problemas e desafios de magnitude assinalável, cujas características fundamentais parecem não se alterar ao longo do tempo. As equipas de TI continuam a ser crónica e sistematicamente confrontadas, vezes sem conta, com as mesmas tarefas de integração, instalação, actualização, *helpdesk*, diagnóstico, reparação/recuperação e inventário de equipamento. Longe de ser trivial ou relegável para segundo plano, a gestão de *desktops* é na realidade uma tarefa exigente e desgastante, constituindo um sorvedouro de consideráveis quantidades de tempo e recursos. Levando em consideração que existem casos onde o crescimento da capacidade da estrutura de TIs se faz de forma semelhante à Lei de Moore, duplicando a cada 12-18 meses [Busch 2004], torna-se visível a real dimensão do problema.

Por estas razões, e apesar da sua menor popularidade no contexto global da investigação em gestão de redes e sistemas distribuídos, a gestão de *desktops* é um objecto de estudo de pleno direito. É nesta linha de pensamento que surge o trabalho associado a esta Dissertação, cujo âmbito incide precisamente sobre as questões relacionadas com a gestão de *desktops* e que dá continuidade a trabalho anteriormente desenvolvido no Laboratório de Comunicações e Telemática (LCT) do Centro de Informática e Sistemas da Universidade de Coimbra. O trabalho subjacente a esta dissertação (que passaremos a designar por *Projecto IC³* [1]) debruça--se sobre a ineficiência e inadequação dos *desktops* correntes (em termos de manutenção e monitorização) e procura alternativas viáveis para a resolução dos problemas que os assolam, sem com isso sacrificar as vantagens decorrentes do seu uso. Procura-se assim desenvolver uma plataforma que conjugue alguns dos aspectos mais interessantes do PC (*Personal Computer*) clássico e de paradigmas alternativos como o *thin client* e o *network computer*.

1.2 Objectivos

Quando analisado em detalhe, o computador pessoal que tipicamente se encontra nos postos de trabalho (PC compatível IBM) revela assimetrias e deficiências a vários níveis – desde o projecto até à implementação. Estes computadores formam, no seu conjunto, uma amálgama pouco coesa que mistura tecnologias recentes (por vezes em demasia!) com componentes obsoletos (muitos dos quais mantidos em nome de uma inútil retrocompatibilidade). O PC pode ter evoluído significativamente em termos de performance e características arquitecturais, mas subsistem aspectos em que o conceito se mantém primitivamente similar ao modelo original[2].

Apesar de se terem registado inúmeros esforços no sentido de dotar o PC de funcionalidades de gestão adequadas, nas diversas evoluções/gerações de *hardware* e *software*, o resultado final – na forma do PC contemporâneo – saldou-se num conjunto de alterações com pouca ou nenhuma articulação entre si, que muitas vezes apenas dizem respeito a aspectos específicos, esparsos ou de reduzida relevância quando analisados no contexto global da plataforma. Ainda que esta tendência se tenha recentemente começado a inverter – com o surgimento de tecnologias e iniciativas criadas para promover recursos de gestão realmente eficazes e

[1] O termo *IC³* é um acrónimo de *Integrated Communications and Computing Concept* ou, na forma adoptada para título desta dissertação, *Plataforma Integrada de Computação e Comunicações.*

[2] Pode talvez arriscar-se uma analogia que compare um PC a um automóvel da década de 20 equipado com um *tandem* motor eléctrico/*fuel cell*: uma máquina evoluída, mas dificilmente homogénea.

mutuamente coerentes entre si – estas não deixam de ser adaptações incorporadas numa plataforma que não foi pensada à partida, com a perspectiva da gestão em mente.

Assim, ao invés de enveredar pela tradicional abordagem paliativa, baseada na colagem de mais serviços e funcionalidades para resolver as velhas lacunas das plataformas existentes, o Projecto IC^3 propõe uma solução alternativa, assente num novo paradigma situado a meio termo entre os sistemas centralizados e o cliente-servidor: o *balanced computing.*

Na sua essência, o Projecto IC^3 está alicerçado em torno de dois objectivos-chave:

- Conceber uma alternativa ao PC desktop clássico e uma arquitectura de serviços capazes de proporcionar aos utilizadores uma experiência de uso dos meios computacionais que seja mais completa, mais produtiva e mais gratificante.
- Disponibilizar aos gestores de sistemas os recursos de que necessitam para assegurar o desempenho eficiente e robusto da infraestrutura distribuída sob a sua responsabilidade.

A plataforma desenvolvida no Projecto IC^3 foi assim pensada de raíz tendo em conta os seguintes requisitos fundamentais:

- **Disponibilidade.** Tanto a plataforma a desenvolver como a arquitectura de suporte devem fornecer um serviço de elevada disponibilidade, quando comparadas com as soluções tradicionais. Para o efeito foram consideradas soluções técnicas que permitiram adaptar e/ou adoptar componentes com elevado grau de resiliência e robustez a falhas, quer sejam induzidas ou acidentais.
- **Eficiência e Equilíbrio.** A eficiência, seja ela em termos energéticos, de gestão, arquitecturais ou computacionais, deve ser uma característica a ter em conta, procurando-se um equilíbrio adequado entre os requisitos a endereçar e os recursos considerados necessários para que a arquitectura a desenvolver esteja à altura de lhes dar resposta.
- **Simplicidade.** Seja em termos de gestão ou de utilização, a solução deve primar pela simplicidade q.b., procurando um compromisso entre flexibilidade, ergonomia, usabilidade, transparência e facilidade de manutenção.
- **Custo.** O custo, entendido como o somatório de inúmeras parcelas (equipamento, software, recursos humanos, infra-estrutura de comunicações e muitas outras parcelas), tem forçosamente de ser tido em conta na solução desenvolvida. Para que tal seja possível, será levada a cabo uma identificação dos pontos críticos onde poderiam ser conseguidas reduções efectivas do custo de manutenção, posse e gestão através da adopção de soluções técnicas e metodologias adequadas.

Estes quatro pontos constituiram as linhas de acção assumidas para o Projecto e determinaram, em traços gerais, a grande maioria das opções de concepção que foram tomadas ao longo do trabalho.

1.3 Estrutura da dissertação

Este documento encontra-se organizado em 6 capítulos (Figura 1.1).

No presente Capítulo é discutida a motivação por detrás deste projecto, os seus objectivos e as linhas de acção a seguir para os atingir.

No Capítulo 2 é apresentada uma perspectiva diacrónica da gestão de *desktops* – desde a sua génese até aos nossos dias – procurando identificar os problemas chave ainda por resolver nas soluções correntes.

No Capítulo 3 é discutido o trabalho que tem vindo a ser levado a cabo no LCT, o actual estado da arte e novos modelos emergentes na área da gestão de *desktops*. Nesta perspectiva, é ainda proposto o paradigma de *balanced computing platform* e são apresentados os projectos IC^3 e DOMUS.

No Capítulo 4 são estudados em detalhe os vários aspectos da arquitectura IC³, sua concepção, desenvolvimento e integração, passando ainda pela arquitectura de *hardware*, *software de sistema,* serviços e aplicações suportadas. A ordem de abordagem dos temas segue deliberadamente uma perspectiva *bottom-up*, do *hardware* ao *software* aplicacional.

O Capítulo 5 é dedicado aos aspectos da plataforma IC³ especificamente relacionados com o suporte de *disconnected computing* e com a operação em modo *offline*. Estes aspectos foram incorporados na plataforma para dar suporte a postos de trabalho móveis. À semelhança do Capítulo 4, também aqui é seguida uma abordagem *bottom-up*, começando a discussão ao nível do *hardware* e terminando com aspectos relacionados com serviços e *software* de sistema.

O Capítulo 6 conclui esta Dissertação, analisando os resultados obtidos, identificando as contribuições daí decorrentes e discutindo as perspectivas de trabalho futuro.

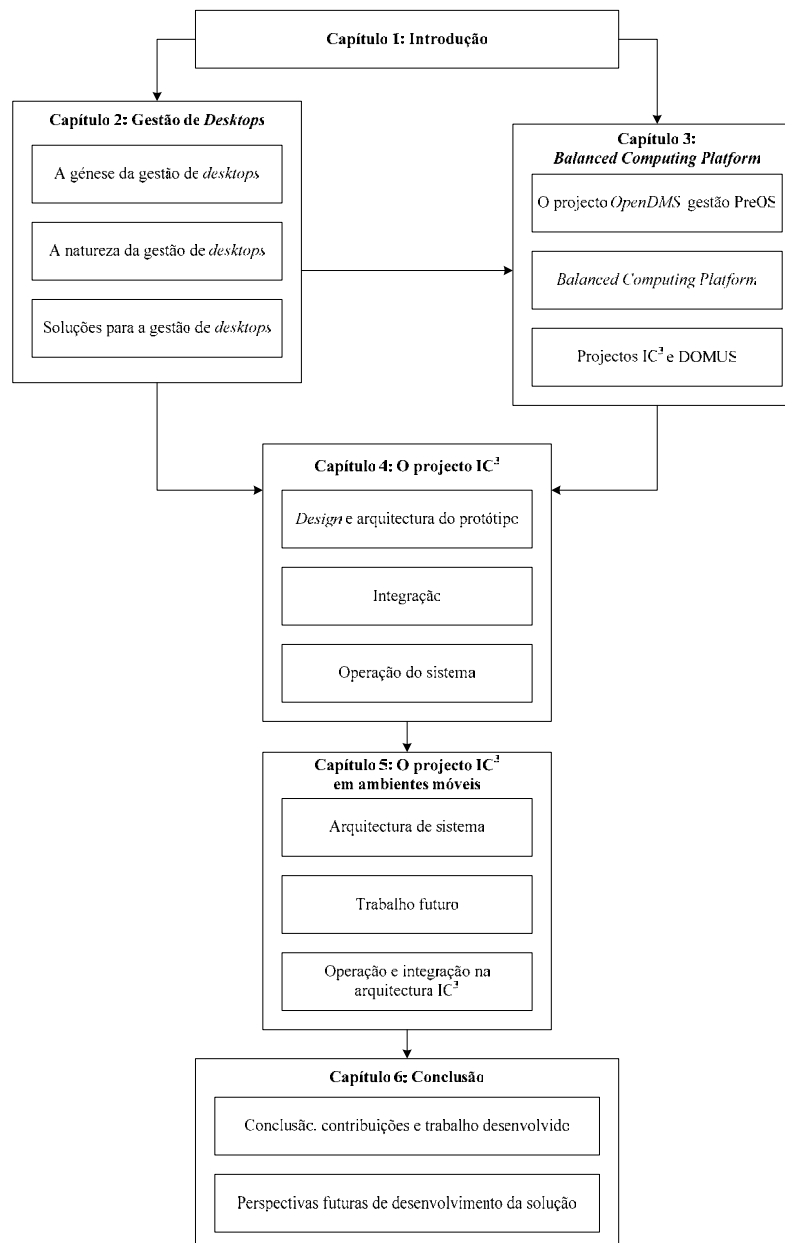



Figura 1.1 – Estrutura da Dissertação

2. Gestão de *Desktops*

Neste capítulo a gestão de *desktops* é contextualizada *n*uma perspectiva histórica, dando a conhecer as suas origens e problemas e discutindo as diversas abordagens que foram sendo propostas, ao longo do tempo, para colmatar esses problemas. Procura-se assim fornecer uma perspectiva diacrónica da evolução dos vários paradigmas de gestão e dos vários paradigmas de uso dos meios computacionais, já que estes dois temas estão intimamente interligados.

O capítulo encontra-se estruturado em cinco Secções distintas:

Na Secção 2.1, dedicada à era da *mainframe*, recorda-se a época dos sistemas centralizados, com especial ênfase nos aspectos mais relevantes do seu uso e operação.

Na Secção 2.2 aborda-se a chegada do PC e a consequente mudança de paradigma, analisando-se os motivos subjacentes à mudança dos sistemas centralizados para o modelo cliente-servidor a fim de compreender de que modo o computador pessoal constituiu uma tecnologia disruptiva que rompe com os modelos previamente estabelecidos.

Na terceira Secção discutem-se as promessas do modelo cliente-servidor, lançando-se um olhar crítico sobre o paradigma cliente-servidor com o objectivo de identificar lacunas e aspectos específicos onde não se concretizaram os benefícios incialmente previstos.

A Secção 2.4 apresenta algumas tentativas da indústria para partir em busca de novas soluções para a gestão de PCs, identificando e enumerando as tecnologias desenvolvidas para dar resposta aos problemas da gestão de *desktops* e do próprio PC enquanto plataforma.

A Secção 2.5 conclui o Capítulo.

2.1 A era da *mainframe*

Na época em que o paradigma centralizado era dominante, o modelo da infraestrutura de TIs disseminado pelas organizações caracterizava-se por uma hierarquia em que o lugar de topo era ocupado por um sistema de grande porte, vulgo *mainframe*, encontrando-se nos extremos inferiores dispositivos de *interface* constituídos por um teclado e um monitor CRT, com poder computacional nulo ou quase nulo – os chamados *terminais*.

Nestes sistemas de grande porte o *hardware* era – e ainda é – concebido e construído tendo em mente aquilo que poderemos designar por *critical-design attitude*: incorporando redundância e mecanismos de protecção eficientes de modo a satisfazer requisitos exigentes de fiabilidade, disponibilidade e resiliência. O mesmo se constata com o *software* de sistema, desenvolvido com uma filosofia idêntica, que privilegia a estabilidade e maturidade sobre a acumulação descontrolada de *features*. Citando o Dr. Barry Feigenbaum, engenheiro de *software* senior da IBM, "seria necessário um acto de Deus para mudar uma linha de código do escalonador de processos do OS/390[3]" [Halfhill l998].

Como consequência imediata da especial atenção que é dada a todos os aspectos da sua concepção, operação e gestão, é comum observar nas *mainframes* modernas *uptimes* de 99.99% ou mesmo 99.9999% [Shapiro 2001]– o equivalente a ter entre 5 a 53 minutos de suspensão de serviço por ano, com MTBCFs (*Mean Time Between Critical Failures*) que podem atingir 20 ou mesmo 30 anos. Esta será uma das principais razões que justificam que ainda hoje muitas organizações adquiram e mantenham *mainframes* em operação.

Indistintamente do contexto temporal – passado ou presente – para os sistemas centralizados a disponiblidade constitui de facto a prioridade absoluta. Se uma *mainframe* com um milhar de utilizadores conectados sofrer alguma falha crítica, não se pode efectuar uma reinicialização desta e continuar como se nada fosse, pois todos os dados e programas estão armazenados nesse mesmo sistema, que concentra em si a totalidade da capacidade computacional existente.

Assim sendo, as interrupções de serviço não devem ser um imprevisto, mas sim um evento agendado como parte integrante de um programa de manutenção preventiva pensado ao pormenor, pois caso ocorra uma falha inesperada os utilizadores serão os primeiros a exigir ser informados da causa que a motivou e a fazer pressão para que esta seja rapidamente solucionada. Esta atitude por parte dos utilizadores, misto de exigência, apreensão e desassossego, é consequência de um condicionamento comportamental específico do paradigma centralizado assimilado pelos indivíduos através da *praxis* quotidiana do uso das TIs, sendo a perspectiva da total e incompleta impossibilidade de trabalhar em caso de falha a sua principal causa (é preciso não esquecer que um terminal é uma entidade passiva, de nada servindo se não estiver conectado a um sistema central operacional).

2.2 A chegada do PC e a mudança de paradigma

No início da década de 70, o progresso da tecnologia LSI (*Large Scale Integration*) e a consequente invenção do microprocessador veio possibilitar o desenvolvimento dos primeiros microcomputadores. Não passando inicialmente de meras curiosidades, estes evoluiram rapidamente em termos de desempenho e capacidades, de tal forma que os grandes actores da indústria das TIs da época se viram obrigados a tomar medidas no sentido de apanhar uma fatia deste mercado em franco crescimento. A IBM não foi alheia a este facto e, de forma discreta, levou a cabo o desenvolvimento de um sistema pessoal que seria lançado no mercado em 1981: o IBM PC [IBM 1981]. Supreendentemente, e apesar da sua arquitectura algo conservadora, mesmo para a época, o IBM PC é um sucesso quase imediato.

Curiosamente, esse sucesso é em parte devido a um relacionamento inesperado com a tecnologia que viria a suplantar mais tarde. Os PCs fizeram a sua tímida aparição no meio

[3] o OS/390, actualmente apelidado z/OS, é o sistema operativo das *mainframes* z/Server da IBM.

corporativo ocupando um lugar ao lado dos terminais 3270 (Figura 2.1), utilizados para aceder às *mainframes* IBM onde se encontravam os programas e dados críticos das organizações. No entanto o PC original não era uma máquina de dimensões contidas, e o espaço que ocupava nas mesas de trabalho (onde já se encontravam os terminais) dificultava a vida aos utilizadores e condicionava, de certa forma, a sua aceitação.

IBM PC Terminal 3270

Figura 2.1 – Um Vislubre do Passado

A solução para este problema surgiu pela mão da empresa *Technical Analysis*, ao conceber e comercializar o adaptador IRMA 3270. Quando instalado num *slot* de expansão, este adaptador permitia a um IBM PC conectar-se a uma *mainframe*, dispensando a necessidade de um terminal dedicado e tornando o PC numa máquina *corporativamente correcta*. O sucesso desta solução foi tal que a IBM chegou a comercializar o modelo IBM 3270 PC, que incluia o adaptador IRMA 3270 instalado de fábrica.

Longe das grandes corporações, nas organizações de pequena e média dimensão, a aparição do computador pessoal vem tornar acessíveis os benefícios do processamento informático de dados outrora reservados aos detentores de orçamentos compatíveis com a manutenção dos sistemas de grande porte. Desde o processamento contabilístico ao tratamento de texto, o computador de secretária – mesmo operando isolado e usando disquetes como único meio de troca de dados com outros sistemas – constitui uma reconhecida mais-valia (*cfr. caixa*).

Com a disseminação de PCs pelas organizações, o passo seguinte seria, naturalmente, ligá-los entre si por forma a viabilizar a partilha de dados e/ou programas através de uma LAN (*Local Area Network*). Apesar de existirem soluções para ligar PCs em rede desde meados da década de 80, apenas no final desta década a tendência ganha *momentum*, com o advento dos paradigmas cliente-servidor e *workgroup computing*.

A mudança de paradigma – que se vinha anunciando desde o dia que os PCs foram aceites na organizações pela primeira vez – concretiza-se finalmente, dando os terminais lugar a *desktops* interligados por uma LAN e as *mainframes* a um conjunto de sistemas desempenhando um papel central de fornecedores de serviços e/ou repositórios centrais de dados, podendo estas estruturas ir da típica rede *workgroup* com meia dúzia de PCs até às gigantescas *Intranets* empresariais compreendendo uma míriade de recursos *enterprise-wide* críticos e vitais.

O computador pessoal: uma invenção dos utilizadores

Curiosamente, o conceito do computador pessoal surgiu quase por acaso quando um fabricante de terminais, a *Computer Terminal Corporation*, desenvolveu o terminal programável *Datapoint 2200* (possuindo uma CPU de 8-bit construída totalmente com lógica discreta TTL e que está na génese do primeiro microprocessador de 8-bit, o intel 8008) que dispunha da capacidade de carregar as emulações de terminal a partir de fita magnética.
O que acabou por suceder é que os clientes acharam outros usos para este terminal programável e desenvolveram programas de âmbito geral, que eram carregados no lugar das emulações de terminal e que iam desde a simples contabilidade à gestão financeira, permitindo utilizá-lo de forma autónoma sem necessidade de o ligar a um sistema de grande porte.

A arquitectura cliente-servidor, assente na descentralização do poder computacional, tomara o lugar do velho modelo centralizado (Figura 2.2) com a promessa de ganhos substanciais no TCO (*Total Cost of Ownership* [Gartner 1997] [Heilbronner 1997]). Conforme refere [Berson 1992], a adopção do modelo cliente-servidor tem um efeito positivo em termos de TCO na medida em que beneficia um conjunto de variáveis que o afectam directa e/ou indirectamente – para citar, "*to reduce software maintenance costs, increase software portability, boost the performance of existing networks [...] increasing developer's productivity and shortening the development lifecycle*".

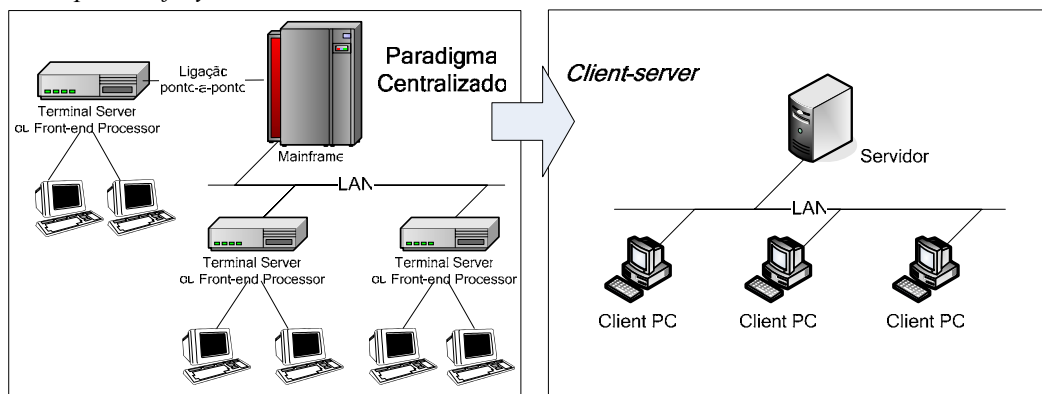



Figura 2.2 - Do Modelo Centralizado para o Modelo Cliente-Servidor: Antes e Depois

2.3 A promessa do paradigma cliente-servidor

Decorrido algum tempo, começou-se a constatar que o modelo cliente-servidor não cumprira com algumas das suas promessas, nomeadamente nos aspectos relativos à redução do TCO. Para identificar as reais causas deste insucesso é necessário voltar um passo atrás, procurando vislumbrar, num contexto mais amplo, a direcção em que a evolução tecnológica nos tem conduzido.

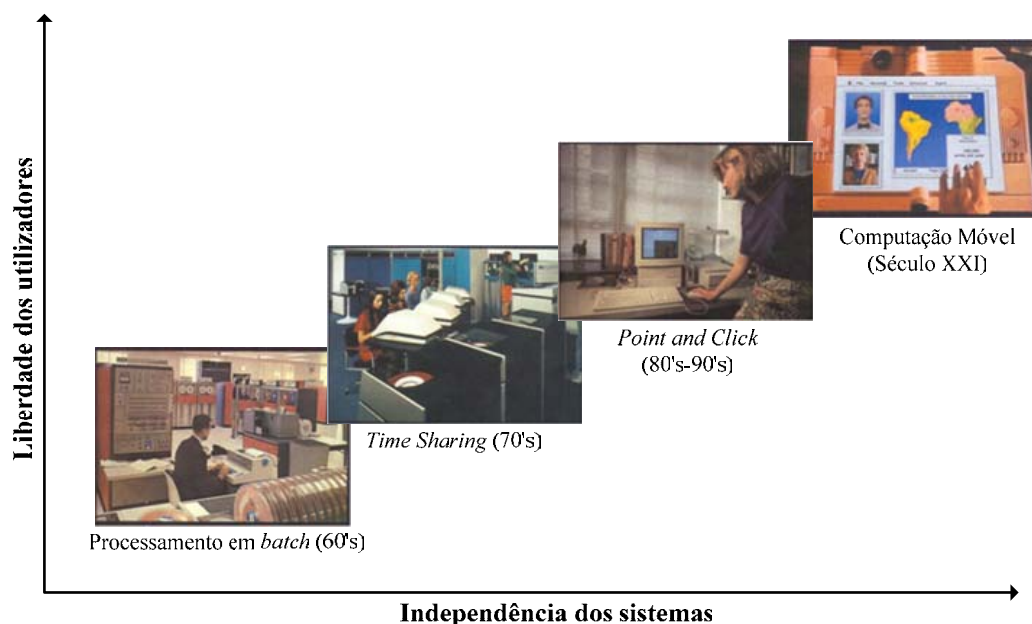



Figura 2.3: A mudança de paradigma na utilização dos meios informáticos

Desde o surgimento dos primeiros sistemas de uso geral até aos dias de hoje assistimos a mudanças profundas associadas à transição entre distintos paradigmas do uso dos meios computacionais (Figura 2.3). Olhando para o caminho percorrido desde o *batch processing* até ao *point and click*, passando pelo *time-sharing* e terminando na emergente era da

mobilidade, torna-se evidente que o triunfo da Lei de Moore [Moore 1965] teve como principal consequência o crescente *empowerment* do utilizador, graças à disponibilidade de maior poder computacional por posto de trabalho. Do simples papel de ferramenta, os meios computacionais evoluiram, ao longo do tempo, para a função de colaboradores, à medida que o *status* do utilizador transitava da subserviência (na era do *batch processing*) para a liberdade total (*cfr. caixa*).

Contudo, a liberdade dos utilizadores colide com o interesse das equipas de equipas de gestão de sistemas e a grande promessa do modelo cliente-servidor – um computador por posto de trabalho – trouxe consigo uma mudança profunda na natureza dos problemas com que estas equipas estavam habituadas a lidar.

Paradigmas de Uso dos Meios Computacionais e Autonomia dos Utilizadores

Cada paradigma possui um conjunto de características distintivas no que diz respeito à relação entre os utilizadores e os meios computacionais:

	Batch	***Time-sharing***	***Desktop***	**Móvel**
Década	1960s	1970s	1980s-90s	2000s - ?
Tecnologia	*Medium Scale Integration* (MSI)	*Large Scale Integration* (LSI)	*Very Large Scale Integration* (VLSI)	*Ultra Large Scale Integration* (ULSI)
Localização dos utilizadores	*Data processing center*	Sala de terminais	Secretária de escritório e/ou doméstica	Móvel
Objectivo	Calcular	Aceder	Apresentar	Comunicar
Actividade dos utilizadores	*Punch & Try* (submeter)	Interactiva	Ver e clicar (guiar)	Pergunta-resposta (delegar)
Interconexão	Periféricos	Terminais	*Desktops*	PDAs, *notebooks*, etc

- Na época do ***batch processing*** os utilizadores trabalhavam em função da máquina, numa relação de dependência e subserviência. O poder computacional era escasso e disponível apenas a uma elite científica ou corporativa, sendo todas as tarefas organizadas de acordo com a conveniência da máquina – nesta época os trabalhos eram submetidos em lotes (*batches*) cuidadosamente escalonados para eliminar tempos mortos e cuja dimensão deveria ser suficientemente grande para justificar o custo de processamento. O acesso e operação destes sistemas eram feitos a partir do *data processing center*, não havendo grande necessidade de interligação dos sistemas entre si, dada a sua escassez e especificidade.
- Com o ***time-sharing***, o aparecimento dos ambientes de operação multitarefa/multiutilizador e a deslocalização dos postos de trabalho (com a disseminação dos terminais) deu-se o primeiro passo para a independência dos utilizadores. Os utilizadores moveram-se para as salas de terminais e passaram a poder utilizar os sistemas de modo interactivo e em termpo real, normalmente para aplicações de natureza transaccional. Para maximizar a eficiência do aproveitamento dos recursos computacionais procurava-se partilhar os sistemas pelo máximo possível e racionalmente comportável de utilizadores. Este paradigma seria impensável sem as redes de dados, sejam sob a forma de ligações dedicadas ou *dial-up* utilizadas para ligar os terminais ao sistema central – foi também nesta época que surgiram que as primeiras ligações entre sistemas distintos para troca de informação.
- Com o paradigma do ***point-and-click***, possibilitado pela aparição de computadores pessoais autónomos com capacidades de processamento adequadas, normalmente ligados a uma rede e dotados de ambientes de operação gráficos WIMP (*Windows, Icons, Mouse and Pointer*) os utilizadores ganharam autonomia e começaram a tratar os meios informáticos como uma ferramenta. Esta foi a era da democratização dos meios informáticos, do modelo cliente-servidor, das primeiras LANs (*Local Area Networks*) e do software de produtividade pessoal, com os computadores a surgirem nas secretárias e a assumirem-se como ferramentas.
- Com a emergência da era da **computação móvel** prevê-se que os meios computacionais evoluam para a função de colaboradores, passando os utilizadores a dispor de autonomia total e a interagir com os sistemas para orquestrar e organizar tarefas em vez de apenas emitir ordens.

Cada era da computação é o reflexo de um conjunto de factores que vão desde a velocidade dos sistemas informáticos contemporaneos à variedade de informação que pode ser processada passando ainda pelas formas de interligação entre os sistemas, o tipo de *software* utilizado ou o modo como os utilizadores interagem com os meios ao seu dispôr.

Uma aproximação *behaviourista* à questão da *cultura das TIs*

A cada paradigma tecnológico encontra-se associado um determinado conjunto de comportamentos e reacções resultantes da dinâmica da interacção entre utilizadores e o sistema. Quando na secção 2.1 se discutiu a natureza da reacção-tipo dos utilizadores em caso de falha da *mainframe*, observou-se que estes tinham desenvolvido um conjunto de comportamentos específicos ao manifestar particular interesse e preocupação pelo funcionamento do sistema – é quase como se existisse um perfil típico do utilizador de uma tecnologia.

Em relação aos gestores sucede algo semelhante. Durante a década de 70, no auge do monopólio das sistemas de grande porte massivamente centralizados, o local de trabalho por excelência dos gestores de sistemas era o *Data Processing Center*, onde estes profissionais qualificados levavam a cabo a quase totalidade das suas funções, dedicando a sua atenção ao cuidado e manutenção do estado da *mainframe* sem devotar grandes preocupações ao o que sucedia nos níveis inferiores da hierarquia da infraestrutura (para esse propósito havia gente menos qualificada ao dispor) – atitude que favorecia a existência de um certo grau de isolamento e distanciamento por parte destes em relação ao exterior e aos utilizadores, contribuindo para que adquirissem em alguns casos um estatuto de quasi-divindades tecnológicas (passe o abuso de linguagem).

No caso do paradigma associado ao modelo cliente-servidor existe também uma "cultura prática das TIs" baseada no princípio de que, se algo corre mal, *os técnicos limpam o sistema e reinstalam tudo* – outros, como é o caso de alguns utilizadores, vão ignorando os sintomas até os sistemas ficarem inoperantes antes de reconhecer que têm um problema. Não só a atitude generalizada dos utilizadores tende a ser mais despreocupada como, em alguns casos, pode raiar o laxismo.

Ao contrário do que sucedia nos sistemas centralizados, os gestores são obrigados a fazer deslocações frequentes aos postos de trabalho para resolver problemas o que muitas vezes fomenta entre os utilizadores uma falsa noção de total disponibilidade por parte destes para resolver qualquer insignificância.

A mudança de mentalidades e hábitos instalados, contrariando a cultura instalada, constitui muitas vezes um dos maiores desafios a enfrentar na hora de atacar o problema do *desktop management.*

Na grande maioria dos casos, os computadores que se encontram em cada posto de trabalho são PCs compatíveis IBM cuja capacidade de processamento atingiu, com o passar do tempo, níveis idênticos ou superiores aos das *mainframes* do passado. Esta proeza foi obtida graças a um conjunto de opções arquitecturais que privilegiaram os factores custo e funcionalidade em detrimento da fiabilidade, favorecendo a adopção de ciclos de desenvolvimento de produto mais curtos. Se compararmos a filosofia de desenvolvimento das *mainframes* com a dos PCs (considerados "*the most crash-prone computers ever built*" [Halfhill 1998]) torna-se evidente que estamos perante dois sistemas com concepções completamente distintas.

Assim, ao eliminar os velhos terminais robustos e *stateless*, passou a ser necessário gerir dezenas, centenas ou mesmo milhares de PCs substancialmente mais complexos, constituindo cada um deles uma entidade autónoma com inúmeros pontos de falha que resultam tanto das características intrínsecas da sua arquitectura como da maior liberdade que passa a ser dada ao utilizador. Em cada PC existe um sem-fim de fragilidades que podem ser transformadas em falhas por um utilizador mais curioso ou aventureiro (*cfr. caixa*).

No mundo das *mainframes*, quando ocorre uma falha crítica a empresa proprietária do sistema contacta o fornecedor para exigir explicações (isto quando o próprio sistema não toma essa iniciativa por si graças a mecanismos automáticos de *reporting*, fazendo o *upload* dos *logs* e outras informações relevantes do sistema para o fabricante). Em contraste, quando um *desktop* falha, o administrador de sistemas nem sequer é notificado da ocorrência por parte dos utilizadores, muito menos o fabricante do *hardware* ou quem desenvolveu o *software*, seja ele de sistema ou aplicacional. Não podemos esquecer que no mundo dos PCs, o *hardware* e o *software de sistema* têm na imensa maioria das vezes proveniências diferentes (devido a uma relação de *loose-coupling* entre os respectivos fornecedores), não se podendo falar de um único "fornecedor de sistema" excepto nas situações em que a organização contrata um pacote de serviços de *lifecycle management* que inclua os componentes e a respectiva assistência e gestão, independentemente da sua proveniência.

Como resultado desta situação, calcula-se que o preço de adquisição um PC corresponda a uma fracção relativamente pequena do seu TCO [Gartner 1997][Busch 2004], quando comparada com os custos acumulados da manutenção do sistema, treino dos utilizadores e *helpdesk* (Figura 2.4). A operação e manutenção do parque de PCs consome frequentemente mais recursos humanos que a gestão conjunta dos servidores, da infraestrutura de

comunicações e dos serviços telemáticos. Mesmo tendo em conta que grande parte das tarefas de *helpdesk* são desempenhadas por técnicos menos especializados ou de *tier-3* (regra geral mais baratos), o seu peso nos custos totais da infraestrutura informática não é, de modo algum, negligenciável. O modelo cliente-servidor tem afinal um custo oculto, capaz de ameaçar seriamente o ROI (*Return Of Investment*) das TIs.

Apesar das suas indiscutíveis vantagens, a verdade é que o advento da era do *desktop* tornou a administração de sistemas numa tarefa consideravelmente mais complexa. A descentralização da infraestrutura de TIs acarretou uma redefinição das prioridades das equipas de gestão que passaram a estar focadas em três aspectos-chave críticos: gestão de inventário, redução do *downtime* e minimização das intervenções *in situ.*

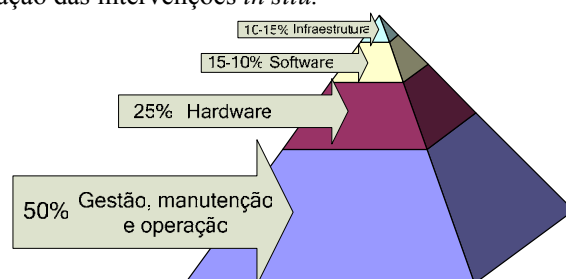



Figura 2.4 – Os Quatro Componentes do TCO do PC [Busch 2004]

2.4 Em busca de Soluções para o Dilema da Gestão de PCs

2.4.1 Revendo a Concepção do PC

São de registar diversas iniciativas da indústria que visaram precisamente repensar alguns dos aspectos mais frágeis do PC e definir normas adequadas para a gestão distribuída de parque de PCs. Muitas dessas iniciativas tiveram por objectivo conceber PCs mais simples, com menos pontos de falha, energeticamente mais eficientes, livres da pesada herança de normas obsoletas, pensados ergonomicamente, pequenos e de aparência mais agradável. É por exemplo o caso do IBM *PS/2 Energy Desktop* [IBM 1993]; das normas *MicroATX* e *FlexATX* da Intel [Intel 1997][Intel 1999], das especificações *ITX e Mini-ITX* da VIA [VIA 2001] [VIA 2001a][VIA 2002] e dos *PC Design Guides* da Intel e Microsoft [Intel/Microsoft 1998].

IBM PS/2 *Energy Desktop*: um PC diferente dos outros

Vista de conjunto, com LCD TFT VGA de 10,4" | Painel frontal, expondo FDD e baías PCMCIA | Painel traseiro, expondo portas RS-232, Centronics, VGA e duas baías PCMCIA | Adaptadores PCMCIA 3270 e Ethernet

Anunciado pela IBM em 1993, o sistema PS/2 Energy Desktop foi uma das primeiras tentativas de repensar o PC, procurando reconciliá-lo com as questões da ergonomia, ambiente e design. Baseado num processador IBM 486SLC2 (uma versão *low-power* do Intel i386SX desenvolvida pela IBM com um multiplicador de frequência de 2x) operando à cadência de 25/50MHz, possuía um chassis compacto (Altura: 63,5mm, Largura: 305mm, Profundidade: 305mm) e dispunha de 4 *slots* PCMCIA para adição de periféricos e opções, tais como discos rígidos de 1,8" em formato PCMCIA III (com 105MB de capacidade) ou mesmo adaptadores 3270 PCMCIA.
Proposto com um teclado compacto dotado de *trackpoint* integrado e um monitor TFT de 10,4", em opção, o IBM PS/2 E (como também se designava) era radicalmente diferente do PC típico da época e estava, de facto, muito à frente do seu tempo. Equipado com uma fonte de alimentação de 35W, em conjunto com o TFT proposto e os mecanismos de gestão de energia de que dispunha podia operar de forma similar a um portátil chegando a não ultrapassar os 20W/hora em modo *sleep*. Em modo de operação normal consumia aproximadamente cinco vezes menos energia que um PC típico da época. Silencioso, ergonómico, pequeno e expansível, o PS/2 "E" era um PC que não fazia compromissos de género algum, possuindo ainda chassis quase totalmente reciclável (a excepção era, curiosamente, o próprio logótipo da IBM no painel frontal).

2.4.2 *Thin clients* e *Network computers*

Outras inciativas foram ainda mais longe, propondo novos paradigmas. Os *thin clients* e os *network computers* [Blundon 1997] surgem neste contexto como modelos vagamente inspirados na era das *mainframes*, onde o terminal era a entidade que permitia ao utilizador comunicar com a máquina central que albergava dados e programas. Dotado de pouca ou nenhuma "inteligência", e sendo extremamente simples, o terminal tinha um reduzido potencial de falha, atingindo um MTBF de 170000h sem grandes dificuldades, aproximadamente uma ordem de grandeza acima do MTBF de um PC [WYSE 2001].

Com os *thin clients* e *network computers* a ideia é essencialmente a mesma: recolocar programas e dados num ponto central, simplificando e robustecendo simultaneamente os pontos terminais. A grande diferença é que o *Network Computer* se baseia num paradigma de *network-centric computing*, com a possibilidade de descarregar e executar aplicações localmente (na maioria dos casos, com recurso a uma máquina virtual Java local), enquanto um *thin client* clássico, que corresponde ao paradigma do *server-centric computing*, se comporta com uma extensão do *display* remoto de um servidor (Figura 2.5).

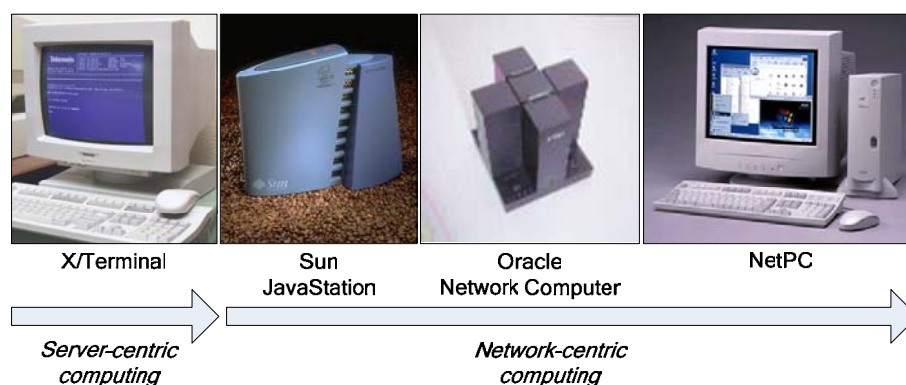



Figura 2.5 – *Thin Clients* e *Network Computers*: Album de Família

Ao contrário do que muitos defendiam, quase não existia benefício imediato em termos de custos decorrente da adopção deste tipo de tecnologias, excepto no respeitante aos subtotais derivados dos tempos de instalação dos postos de trabalho, como consequência da própria natureza dos *thin clients*, em que o *Rapid Application Deployment* é uma característica intrínseca (Tabela 2.1). Um NC ou X/Terminal custava fequentemente o mesmo que um PC e a própria natureza do paradigma implicava um maior investimento na infraestrutura de rede e serviços, visando maior fiabilidade e disponibilidade (o que é característico dos sistemas centralizados). As vantagens existiam sim a médio-longo prazo, e derivavam de aspectos como as reduções nos custos de licenciamento de software e no TCO por posto de trabalho.

	PC em Rede	***Thin Client***
Unpacking e instalação de SO	30 min	5 min
Instalação de Aplicações	1 h	5 min
Configuração de Rede	10 min	5 min
Tempo total dispendido para 1 posto de trabalho	1h40min	15 min
Tempo dispendido para 100 postos de trabalho	24 homens-dia	4 homens-dia
	Diferença: 20 homens-dia	

Fonte: Departamento de Administração de Sistemas da Associação de Informática da Região Centro.
Estimativa aproximada não considerando taxas parciais de produtividade em ciclo de horário de 7h por dia. Tempos para instalação do SO e aplicações no PC em Rede calculados para instalações *unattended* com recurso aos *Remote Installation Services* da Microsoft.

Tabela 2.1 – Estimativa do Tempo de Instalação por Posto de Trabalho

X/Terminal NCR/ADDS 3533
Arquitectura de um *thin client* de primeira geração

O *copyright* da *motherboard* deste terminal *NCR/ADDS X-Station 3533* data de 1993, encontrando-se dotado de um CPU RISC *superescalar* de fabrico *Intel* – o i960CA, cuja primeira geração remonta a 1989. Desenvolvido especificamente para aplicações na área dos sistemas embebidos, o i960CA baseia-se numa arquitectura *load-store,* dispondo de um bus de dados de *32bit* e uma cache de instruções de primeiro nível com 1KB (não dispõe de *cache* de dados). A 25 Mhz, operando com um *clock* externo de 1X, a unidade de controlo de *bus* é capaz de efectuar transferências em *bursts* de 100MB/s. Embebido no *chip* existe também um controlador *DMA* de 4 canais capaz de proporcionar transferências com taxas de pico da ordem dos 45MB/s a 25Mhz.

A conexão com a memória é feita através de um ASIC NCR que contém uma FIFO e a lógica de *interface*, em conjunto com 2 PALs para gerar os sinais de controlo do *bus*. O controlo do *bus* interno da *motherboard* é feito por um ASIC NCR que engloba a funcionalidade de controlador Multi I/O (portas paralela, série, teclado e rato). A isto junta-se um DAC AT&T (capaz de operar a 67.5Mhz para possibilitar resoluções da ordem dos 1280x1024x8bit a 70Hz) associado a 2MB de *VRAM dual-ported* e um ASIC *i82596* para fornecer a *interface Ethernet (10Mbit* na prática e *20Mbit* em teoria, em modo *full-duplex* não suportado pelo *firmware).*

O *firmware* do sistema encontra-se armazenado em 2 *chips* de memória *Flash*, totalizando 384KB. Acrescenta-se ainda um um SIMM equipado com 8 *chips i28F004* (*Flash EEPROM 512Kbit*) com capacidade total de 4MB que contém o servidor X optimizado para o processador *i960,* um *window manager* minimalista e algumas fontes gráficas.

A fonte de alimentação dissipa o calor por convecção, o que reduz a zero o número de peças móveis e, consequentemente, o nível de ruído emitido.

Estimativas divulgadas pelo *Gartner Group*, datadas de 1995, calculavam que uma companhia com 24 *Terminais X* e uma rede baseada em servidores *Unix* podia poupar cerca de 200.000 USD em 5 anos quando comparada com uma infraestrutura baseada em PCs (Tabela 2.2).

No entanto, *thin clients* e afins não chegaram a atingir massa crítica. Anunciara-se 1997 como o ano do *Network Computer*, mas na sua edição de Abril de 1998 a revista *Byte* confirmava que as vendas destes dispositivos apresentavam números decepcionantes [Pountain 1998].

As razões normalmente apontadas para o insucesso dos *thin clients* estão relacionadas com as limitações tecnológicas da época em que surgiram, com especial relevância para: largura de banda insuficiente; fraco (ou mesmo medíocre) desempenho das arquitecturas; falta de flexibilidade; necessidade de reescrita de um grande número de aplicações para tirar partido do paradigma (caso nos *Network Computers*); e abordagem seguida na maior parte das implementações comerciais, com *hardware* e *software* fechados e proprietários, o que tornava a diversidade de *designs* e OS concorrentes e incompatíveis entre si um problema ainda maior de que pareceria à partida.

Estimativa do TCO por utilizador/ano

	Custo/Unid.	Suporte Técnico	Administração	*User-related*	Total
Win95 desktop PC	2.532	1.704	1.497	4.052	9.784
Thin client	1.896	1.497	771	2.611	6.775

Fonte: *The Gartner Group*. Valores em USD. Custos com infraestrutura de servidores não contemplados.

Tabela 2.2 – Estimativa do TCO por utilizador/ano [Gartner 1997] [Gartner 1997a]

Esses factores comprometeram de modo fatal as aspirações de plataformas como o terminal *X/Windows*, a JavaStation [Sun 1996], o NetPC [Microsoft 1997] e o NC da Oracle [Oracle 2001], e depois da euforia inicial assistiu-se ao progressivo esmorecer da expectativa gerada em torno destas tecnologias, até quase se deixar de ouvir falar nelas.

Curiosamente, a partir do momento em que a má imagem se começou a desvanecer e a tecnologia atingiu o nível necessário para concretizar a promessa encerrada nos *thin clients* (em especial mercê da disponibilidade de largura de banda adequada nas redes locais e de *hardware* mais evoluído) começou-se a assistir ao regresso deste paradigma, ainda que forma discreta e direccionada a nichos específicos do mercado das TI. É actualmente possível encontrar *thin clients* nos catálogos de fornecedores como a Sun (*SunRay*), HP (série TC), ou Fujitsu-Siemens (linha FUTRO), assim como na oferta de empresas exclusivamente especializadas neste tipo de equipamento como a Axis, Neoware, Igel ou WISE, apenas para mencionar algumas. De forma alguma este cenário poderá ser considerado sintomático de um retorno em grande escala dos *thin clients*, devendo antes ser encarado como uma segunda oportunidade – rodeada de menos *hype* e direccionada para os casos em que os clientes estão familiarizados com o conceito e/ou conscientes das vantagens e limitações inerentes.

2.4.3 Os contributos da *Distributed Management Task Force*

Uma terceira linha de iniciativas que merece registo foi conduzida pela *Distributed Management Task Force* (DMTF) [DMTF]. Ao produzir normas como o *Desktop Management Interface* (DMI) [DMTF 1998], o *System Management BIOS* (SMBIOS) [DMTF 2001], a *Web-Based Enterprise Management framework* (WBEM) [DMTF 1999a] e o *Common Information Model* (CIM) [DMTF 1999], a DMTF abriu caminho para o posterior surgimento de iniciativas emblemáticas, como a *Zero Administration for Windows* (ZAW) [Plastina 1997] da Microsoft e o *Wired for Management* (WfM) da Intel [Feldman 1999] [Intel 1999a] [Weller 1999]. Ainda neste contexto, surgiram as primeiras gerações de plataformas especificamente pensadas para a gestão de PCs, tais como o SMS da Microsoft [Microsoft] e o LANDesk da Intel [Intel], normalmente baseadas no recurso a agentes que eram instalados nos PCs, suportando a execução remota de operações de monitorização, gestão de inventário (*hardware*, *software*) e configuração. O administrador servia-se de uma consola de gestão para coordenar a operação dos agentes, recolher e processar a informação recebida.

A principal fragilidade deste modelo, baseado exclusivamente em mecanismos *runtime*, deriva do facto de os agentes de gestão dependerem do bom funcionamento do *hardware* e do sistema operativo do PC. Se ocorre uma falha num *file system* – ou se o sistema operativo local se torna instável – o agente *runtime* torna-se inútil e o problema terá de ser resolvido com recurso a uma intervenção local.

Outro problema que afecta estas plataformas é o deficiente integração ao longo de todo o seu espectro, desde o *hardware* até às APIs e *software* de gestão. A plataforma SMS da *Microsoft*, por exemplo, até à versão 2.0, não suportava a gestão de *thin clients* baseados em *Windows* (essa funcionalidade aparece apenas com o Service Pack 1 da versão 2003, e mesmo assim dirigindo-se exclusivamente a *thin clients* baseados em Windows XPe). Apesar de se anunciarem como compatíveis com diversas normas da indústria, a interoperabilidade entre soluções de origens distintas é mínima ou de todo inexistente, pois quase toda a funcionalidade é baseada em APIs e agentes *runtime* proprietários, fechados (e portanto impossíveis de adaptar conforme os requisitos de cada um, para além das funcionalidades *out of the box*) e dependentes do sistema operativo.

2.5 Conclusão

A noção mais difundida e aceite sobre a gestão de *desktops* assenta no princípio de que a indústria das TIs terá ignorado a situação durante anos, nunca tendo disponibilizado mecanismos de gestão cuja adequação estivesse à altura dos progressos técnicos que entretanto se registaram. Esta ideia possui um indesmentível fundo de verdade, ainda que

tenha sido frequentemente veiculada unicamente com o intuito de servir ocasionais campanhas de promoção de algumas soluções comerciais. É no entanto necessário fazer uma análise honesta das várias tecnologias, soluções e recursos desenvolvidos com o objectivo de dar uma resposta aos problemas de *desktop management* antes de cair na tentação de as classificar, de forma redutora, como um total fracasso.

Desde os *thin clients* – vítimas da excessiva pompa com que foram anunciados e das limitações tecnológicas da época em que surgiram – até às normas do DMTF – tantas vezes deturpadas por implementações parciais ou incorrectas – o cenário geral é caracterizado, por um lado, por uma profunda falta de coesão e integração de soluções e normas e, por outro, por graves lacunas para as quais não existem alternativas capazes. Os vários esforços até agora empreendidos saldam-se por um resultado inglório, mantendo-se inalterados os principais problemas do *desktop management* que, por sua vez, incidem de forma recorrente sobre os mesmos aspectos. Existe mesmo uma analogia do estado actual das coisas que defende que, no que diz respeito aos aspectos ligados à gestão e manutenção, "*a indústria das TIs assemelha-se a dois técnicos sentados na asa de um jacto que estão a verificar o funcionamento de um dos reactores com um microscópio para se assegurarem que tudo está a funcionar correctamente*" [Busch 2004]. Neste capítulo constata-se de facto que foi precisamente a falta de sofisticação, coesão e adequação das ferramentas e soluções de gestão a responsável pelo perpetuar e acentuar dos problemas associados à gestão de *desktops*.

3. Balanced Computing Platform

No Capítulo 2 foram discutidos diversos aspectos de âmbito geral relacionados com a gestão de *desktops*. Essa discussão é complementada neste Capítulo com a apresentação das linhas de investigação que têm sido exploradas pelo candidato, nesta temática específica, no âmbito do seu trabalho de investigação levado a cabo no LCT (Laboratório de Comunicações e Telemática do Centro de Informática e Sistemas da Universidade de Coimbra).

Esta apresentação do trabalho prévio do candidato é essencial para que se possa compreender o contexto que permitiu formular e abordar o problema proposto nesta dissertação.

A Secção 3.1 descreve o Projecto OpenDMS (*Open Desktop Management Solution*) [Cruz 2003]. Este projecto, iniciado quatro anos atrás, focou-se na optimização da gestão remota de PCs, por meio de mecanismos mais robustos de gestão remota (em especial a chamada gestão PreOS) e por uma primeira experiência de aplicação de paradigmas alternativos ao PC clássico, nomeadamente pela reintrodução de *thin clients* e *network computers*.

A Secção 3.2, tomando como referência as ilações retiradas do Projecto OpenDMS e a evolução entretanto registada na gestão de *desktops* em geral, discute o cenário actual, identificando os seus pontos positivos e negativos.

Tomando como ponto de partida essa discussão, a Secção 3.3 propõe um novo paradigma de *desktop*, designado por *Balanced Computing Platform*. Este novo paradigma, propiciado pela emergência de novas tecnologias e tendências de uso às quais o PC clássico terá inevitavelmente de se adaptar, situa-se a meio caminho entre o *network computer* e o PC clássico e propõe-se conciliar algumas das melhores características de cada um destes modelos.

A Secção 3.4 define as linhas gerais do Projecto IC^3. Este Projecto – cujo principal objectivo é construir uma *Balanced Computing Platform* que sirva como prova de conceito – constitui a base do trabalho de investigação subjacente a esta dissertação e será mais detalhado nos Capítulos 4 e 5.

Nessa Secção é também apresentada uma breve referência ao DOMUS (*Domestic Oriented Multiservice Self-Managed Appliance*), um projecto paralelo ao IC^3, com o qual partilha tecnologias e paradgimas, mas focado na construção de *appliances* para ambientes domésticos.

Por último, a Secção 3.5 conclui o Capítulo.

3.1 Trabalho Prévio: o Projecto OpenDMS

3.1.1 Objectivos

Foi o cenário descrito no capítulo anterior que motivou a primeira abordagem do candidato à temática da gestão de *desktops*, quatro anos atrás. Essa abordagem, substanciada no Projecto OpenDMS, tinha por objectivo propor ferramentas e metodologias mais racionais e mais eficazes que as plataformas contemporâneas de gestão de *desktops*, tipicamente baseadas numa aplicação centralizada que controla agentes de gestão instalados nos PCs, de modo a assegurar serviços de monitorização e configuração remota.

Os objectivos traçados para o OpenDMS foram delineados a partir do conjunto de vectores-críticos ilustrado na Figura 3.1.

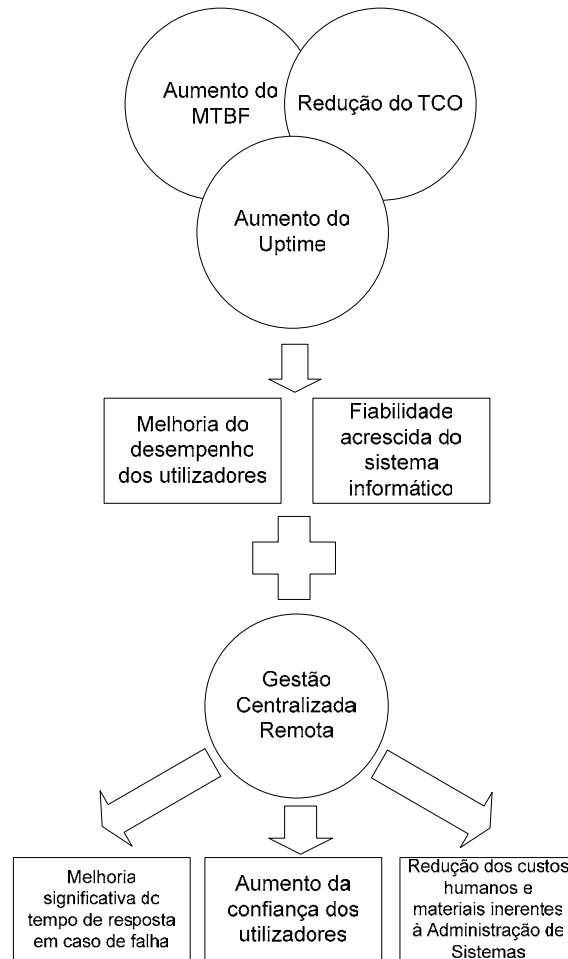



Figura 3.1: Objectivos Genéricos do OpenDMS

De modo a proporcionar um complemento de funcionalidades relativamente às plataformas clássicas, o OpenDMS dá especial atenção a dois recursos essenciais e inovadores: os agentes PreOS (cfr. Secção 3.1.2) e o suporte de modelos alternativos ou híbridos de *fat* e *thin clients*, numa perspectiva de revisão do *desktop* clássico. Adicionalmente, foi também dado ênfase à questão da neutralidade de plataformas.

Isto não esgota as possibilidades da plataforma OpenDMS, que constitui *per se* uma solução completa de gestão distribuída, concebida em torno de normas abertas, ferramentas *opensource* disponíveis sob licença GNU/GPL e que inclui por exemplo agentes *runtime* em muitos aspectos similares aos que se podem encontrar nas soluções tradicionais. No entanto, será sobre esses dois vectores que incidirá a descrição aqui apresentada, dado constituírem os aspectos de inovação mais relevantes do Projecto.

3.1.2 Gestão PreOS e Neutralidade de Plataformas

Uma das inovações do OpenDMS surgiu como resposta a um dos principais problemas das plataformas de gestão existentes há quatro anos atrás: a ausência de mecanismos de gestão capazes de intervir no instante prévio à carga do sistema operativo (estágio PreOS), para fins de execução de procedimentos de diagnóstico e/ou recuperação dos postos de trabalho (no caso de surgirem problemas passíveis de impossibilitar o arranque dos PCs). Os únicos mecanismos deste tipo conhecidos na altura limitavam-se a permitir a (re)instalação completa de sistemas operativos e/ou clonagem de ambientes de trabalho em redes com PCs prontos a instalar.

Na plataforma OpenDMS tal é possível graças ao agente PreOS (Figura 3.2), capaz de intervir durante a sequência de inicialização do PC, logo depois do POST (*Power On Self Test)* e ainda antes da carga do sistema operativo. Neste instante o agente PreOS desempenha diversos procedimentos de gestão recorrendo única e exclusivamente a alguns dos recursos de *hardware* e *firmware* do PC. Ao minimizar a dependência de um determinado sistema operativo, e tendo por alvo o PC *standard* construído com componentes *of-the-shelf*, o agente PreOS cobre a generalidade das plataformas de *desktops*, com excepção de máquinas *Apple* e sistemas com *hardware* mais exótico.

Logo depois dos testes de inicialização (POST), o uso de uma *boot ROM* (*Preboot Execution Environment* (PXE) [Intel 1999b] permite forçar um desvio da sequência normal de inicialização para carregar e iniciar um agente de gestão PreOS (disponibilizado e controlado por um servidor presente na rede local). A máquina pode então ser alvo de operações de autenticação, parametrização, manutenção e monitorização remota, isto é, passa a ser gerível.

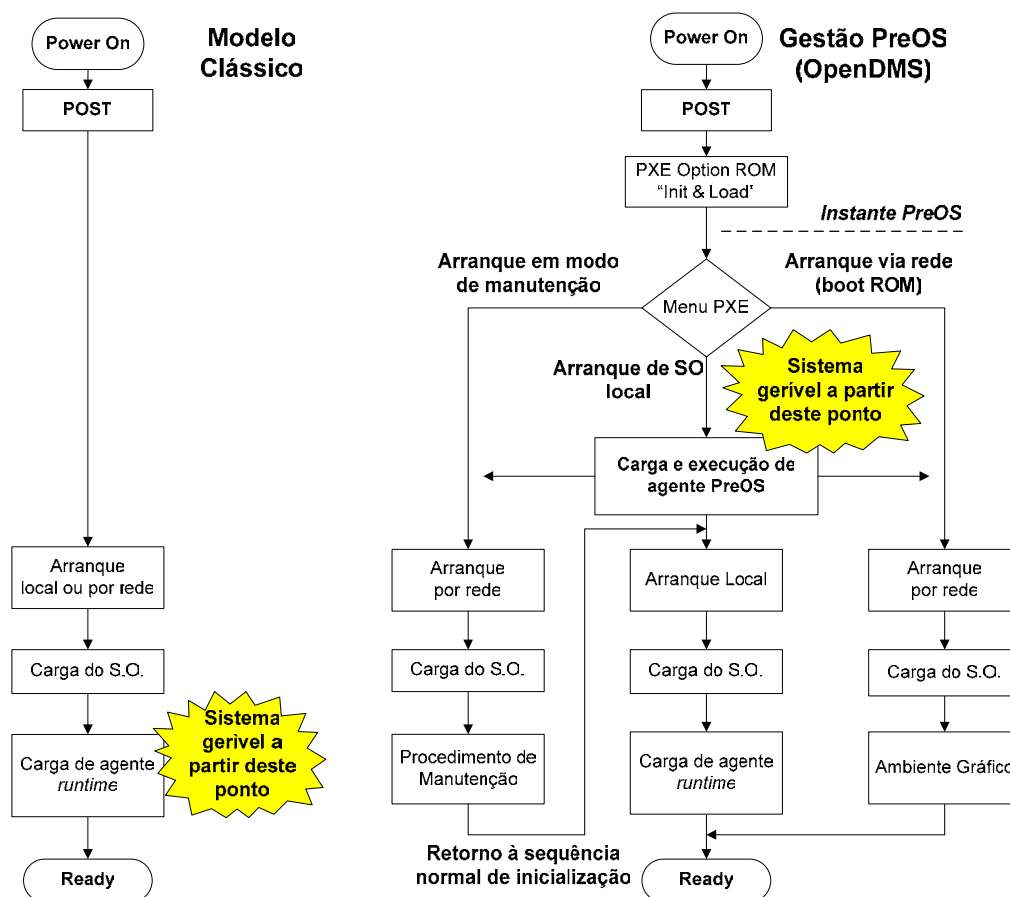
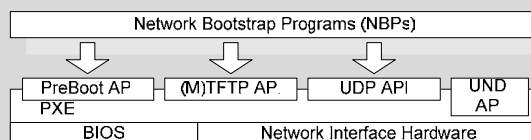



Figura 3.2: Gestão PreOS no OpenDMS

As boot ROMs PXE (*Preboot eXecution Environment*)

Até há algum tempo atrás não existia qualquer forma de carregar e executar um agente de gestão PreOS sem efectuar modificações proprietárias e substanciais no *hardware* do PC. Essa possibilidade surgiu apenas com o advento das *network boot ROMs*, extensões de *firmware* concebidas para permitir o arranque por rede de PCs sob controlo de ambientes de operação de complexidade mínima. Ainda que o objectivo confesso destas *boot ROMs* seja "apenas" a eliminação dos meios de armazenamento locais ou a instalação remota de sistemas operativos, não existe nenhuma razão que impeça o seu uso para outros fins, tais como o carregamento e execução remotamente controlada de um agente vocacionado para a gestão do *desktop*.

As *boot ROMs* de primeira geração, no entanto, não cumpriam os requisitos necessários para suportar um agente PreOS, nomeadamente devido ao uso de protocolos *unicast* da família TCP/IP (tendo em mente a descarga de um imagem a partir de um ponto específico da rede), à total incompatibilidade entre implementações de fabricantes distintos e à inexistência de mecanismos de balanceamento de carga e tolerância a falhas. É neste contexto que surgem as *boot ROMs* PXE. Desenvolvidas no âmbito da *Intel Boot Initiative* [Intel 1998], *circa* 1998, as *boot ROMs* PXE são actualmente suportadas por quase todos os interfaces *ethernet* produzidos para a plataforma PC, incluindo as variantes integradas nos *chipsets*.

A figura mostra com algum detalhe a estrutura modular da especificação PXE. Integrada no *firmware* do sistema sob a forma de uma *option ROM*, esta fornece quatro interfaces de programação aos denominados *Network Bootstrap Programs* (NBPs):

- a API *PreBoot*, que fornece os meios para controlar o ambiente PXE na sua totalidade, desde a (des)activação da pilha protocolar TCP/IP embebida até ao acesso à informação contida nos pacotes DHCP que circularam na fase de inicialização da pilha protocolar da *boot ROM*;
- a API (M)TFTP, que suporta operações de transferência de ficheiros por TFTP ou MTFTP;
- a API UDP, que fornece funcionalidades básicas de I/O sobre UDP;
- e a API UNDI (*Universal Network Device Interface*), que abstrai as especificidades do *hardware* de rede por meio de um conjunto de funções que constituem uma camada de abstração sobre a qual todas as outras APIs funcionam, permitindo assim a criação de *device drivers* universais independentes do fabricante ou modelo da *interface* de rede.

Uma vez ligado o sistema e ultrapassada a fase do POST, a BIOS procura e inicializa todas as *option ROMs* disponíveis. É nesta fase que o controlo é passado ao código PXE, que por sua vez descarrega e executa o agente PreOS. Recorrendo a técnicas de balanceamento de carga e tolerância a falhas suportadas sob *multicast IP,* o PXE permite a existência de vários servidores de arranque a partir de onde o PC cliente pode descarregar o agente PreOS.

Os requisitos do agente PreOS são mínimos: bastam algumas extensões standard do *firmware* do sistema (nomeadamente a *boot ROM*, integrada em praticamente todos os interfaces de rede produzidos actualmente), memória, processador, placa-mãe, interface de rede e fonte de alimentação.

Ao contrário do que sucede com as plataformas de gestão convencionais, mesmo que o sistema operativo não arranque correctamente (seja por avaria de componentes não críticos, tais como placas gráficas ou discos rígidos, seja por desconfiguração do *software*) é possível carregar um agente de gestão remota capaz de proceder a um diagnóstico mais apurado e, nalguns casos, corrigir mesmo as falhas sem necessidade de intervenção *in situ*.

Depois da carga do sistema operativo a presença de mecanismos de gestão é assegurada recorrendo a um agente *runtime* semelhante aos agentes das plataformas convencionais, ainda que substancialmente mais simples, dado que parte das tarefas tipicamente associadas ao agente *runtime* foram já desempenhadas pelo agente PreOS.

A Figura 3.3 apresenta o diagrama de estados simplificado do agente PreOS da plataforma OpenDMS. Uma vez iniciada a execução e estabelecida a comunicação com o servidor de

gestão (a fim de receber directivas para a execução das tarefas seguintes) o agente verifica se o posto de trabalho em questão está autorizado a arrancar.

Em seguida, o agente verifica a integridade do sistema, enviando informação recolhida para o servidor de gestão, em conjunto com a configuração do sistema (obtido por meio da DMI/SMBIOS [DMTF 1998][DMTF 2001] ou por acesso directo ao *hardware*, para efeitos de inventário e complemento ao teste de integridade). Se o estado for considerado satisfatório e a informação de inventário recolhida for coincidente com a armazenada, o agente recebe autorização para prosseguir para o estágio seguinte (autenticação do utilizador) ou, em alternativa, prosseguir com o resto da sequência normal de arranque. Se a execução de algum dos estágios críticos do agente não decorrer como planeado é possível bloquear o sistema.

O sistema operativo a carregar pode ser seleccionado pelo agente PreOS (por exemplo em função do utilizador identificado), sendo assim possível criar *desktops* com múltiplas personalidades: *thin client*, PC Windows ou Linux com sistema operativo local, etc. É também possível, pelo mesmo processo, forçar o carregamento por rede de um sistema operativo *lightweight* para operações de recuperação mais complexas (por exemplo reparação especializada de *file systems*).

Sendo desenvolvido com base numa estrutura modular, a funcionalidade do agente PreOS pode ser extendida por adição de módulos adicionais descarregáveis a partir do servidor. Nesta linha de pensamento desenvolveram-se dois módulos: o *memtest86* e o S.M.A.R.T. (*System Monitoring, Analysis and Reporting Technology*), que fornecem meios para diagnosticar com maior profundidade a memória do sistema e os seus discos rígidos, respectivamente.

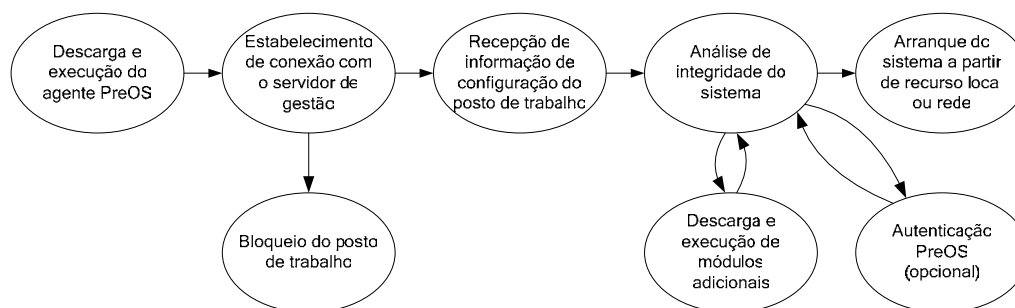



Figura 3.3 – Diagrama de Estados Simplificado do Agente PreOS

3.1.3 A arquitectura OpenDMS

A Figura 3.4 apresenta a arquitectura do cliente OpenDMS (PC gerido). No primeiro nível temos o *hardware* do PC e tecnologias directamente suportadas pelo *hardware*, tais como o *Wake-on-LAN* [Feldman 1999], que possibilita a execução de tarefas de manutenção remota fora de horário de expediente como backups e detecção de vírus, e o *hardware/firmware* de gestão do sistema a baixo nível.

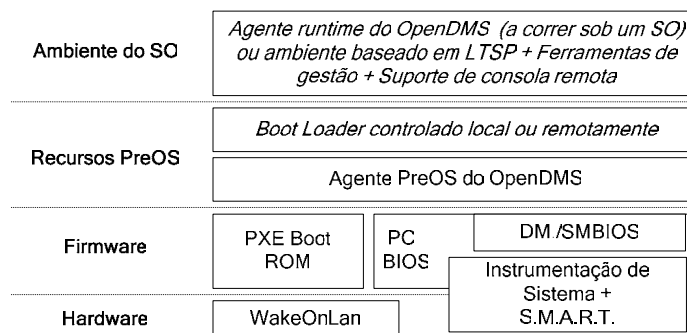



Figura 3.4 – Arquitectura OpenDMS (na perspectiva do cliente gerido)

A segunda camada é constituída pelo *firmware* nativo da BIOS e extensões pertinentes, como é o caso da *boot ROM* PXE e do DMI/SMBIOS. Os PCs modernos disponibilizam directamente todos estes recursos (com a possível excepção das extensões PXE, que são no entanto suportadas por quase todos os interfaces *ethernet* na forma de ROMs opcionais).

O agente PreOS – o componente mais importante da terceira camada – é dinamicamente descarregado do servidor OpenDMS para desempenhar um conjunto de tarefas de gestão num ambiente previo à carga de um sistema operativo. Depois de desempenhar as suas funções o agente PreOS devolve o controlo ao *boot loader* (para carga de um sistema operativo local) ou inicia a carga de um sistema operativo via rede.

Depois de carregado o sistema operativo existem dois cenários possíveis. Para PCs configurados de forma tradicional (arranque a partir de meios de armazenamento de massa locais) a gestão remota é assegurada através do agente *runtime*, coordenado a partir do servidor de gestão. Por outro lado, existe a hipótese de arranque via rede em modo *thin client*, possível graças a recursos disponibilizados pela solução OpenDMS, que providencia uma plataforma para *thin clients* construída sobre componentes PC *standard*, suportando sistemas de ficheiros distribuídos e capacidades de conectividade multi-plataforma com sistemas *X/Windows*, *Citrix Metaframe* [Citrix], *Windows Terminal Services* (WTS/RDP [Microsoft 1999]) e protocolos em modo caractere. Graças ao suporte para acesso a capacidades multimédia e meios de armazenamento de massa locais, esta plataforma cumpre os requisitos de flexibilidade necessários para desempenhar funções que vão do simples *thin client* sem disco ao quiosque multimédia, sem esquecer a possiblidade da existência de configurações híbridas NetPC/PC *standard*.

Do lado dos servidores existem duas entidades fundamentais (Figura 3.5): o Servidor de Gestão de *Desktops* e o Servidor de *Thin Clients*.

O primeiro, implementado sobre um ambiente GNU/Linux, usa o procotolo PXE para descarga do agente PreOS nos PCs a gerir – em associação com o *Trivial File Transfer Protocol* (TFTP) clássico ou com o MTFTP (versão *multicast*). Este servidor é também responsável pelo controlo remoto da execução das instâncias dos agentes PreOS (por exemplo, validando a autenticação de utilizador feita pelo agente) e *runtime*. A consola de gestão usa uma interface web, sendo a informação processada organizada via OpenLDAP [OpenLDAP]. Neste servidor o uso do serviço DHCP é vital para garantir a gestão de uma gama de endereços IP que são distribuídos pelos sistemas cliente de forma dinâmica (por ordem de pedido) ou estática (com base em reserva). Este protocolo é também utilizado em conjunto com o PXE para fornecer um endereço IP aos sistemas durante a inicialização da pilha protocolar IP da *boot* ROM aquando do arranque via rede ou descarga do agente PreOS.

O Servidor de *Thin Clients* exporta para os clientes um ambiente de *thin client* baseado em Linux, suportado por um conjunto de sistemas de ficheiros distribuídos (NFS [RFC 1904], CIFS/SMB [Leach 1996, Microsoft 1988]), protocolos de acesso remoto a dispositivos de armazenamento (NBD [NBD]) e com possibilidade de aceder a meios de armazenamento local, se necessário. Neste contexto são suportados ambientes de trabalho gráfico baseados em *X/Windows* [RFC 1198][Schleifer 1996], WTS/RDP, *Citrix* e VNC (*Virtual Network Computing* [Richardson 1998]). O uso de PXE e TFTP (ou MTFTP) permite descarregar via rede uma imagem auto-encapsulada do sistema operativo do *thin client*, possibilitando o arranque remoto.

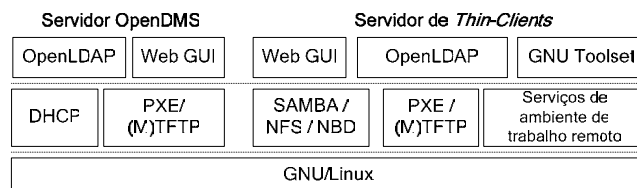



Figura 3.5 – Arquitectura OpenDMS (na perspectiva dos servidores)

Estes serviços podem ser disponibilizados por um único servidor ou um *cluster* de servidores (com serviços distribuídos pelos elementos que o constituem), por razões de desempenho ou disponibilidade. O servidor de *thin clients* é gerido de forma análoga ao servidor de gestão propriamente dito, com interface *web* e um *back-end* suportado por OpenLDAP.

3.1.4 Os *Thin Clients* na Perspectiva da Arquitectura OpenDMS

Alguns anos depois das primeiras gerações de *thin clients*, as duas limitações que mais terão contribuido para o seu insucesso practicamente tinham deixado de existir. Há quatro anos atrás, a largura de banda disponível nas redes locais já satisfazia sem problemas as necessidades dos *thin clients*. O *hardware* usado nos PCs também evoluira bastante, sendo mais do que suficiente para construir *thin clients* sem módulos proprietários.

Um estudo comparativo de performance efectuado para o protocolo *X/Windows* – do qual se reproduzem alguns dos resultados a título ilustrativo (*cfr. caixa*) – demonstrava que se podia suplantar, de facto, os terminais X dedicados recorrendo a PCs, com ganhos de desempenho consideráveis – de uma ordem de magnitude, na maioria dos testes.

Este gráfico mostra parte dos resultados obtidos num teste que serviu para comparar um terminal X dedicado com um *thin client* baseado num PC com processador x86. A bancada de testes estava configurada com o equipamento sucintamente enumerado:

- Terminal X NCR/ADDS 3533 (configuração descrita no Capítulo 2, Secção 2.4.2)
- PC *Intel P54C Pentium 166MHz*, 32MB de memória RAM EDO 70ns, *motherboard* i82430HX, placa gráfica S3 Trio 64V, placa de rede *Intel Etherexpress Pro100*, *linux kernel 2.4.18*, *Xfree86 4.2*
- Servidor com *Intel Pentium Pro 200MHz*, *motherboard Intel PD440FX*, 128MB EDO 70ns, placa gráfica S3 Trio 64, placa de rede *Intel Etherexpress Pro100*, *linux kernel 2.4.18*, *Xfree 4.2.0*)
- Hub 3Com *LinkBuilder TP/8* e *Switch 3Com OfficeConnect 16*

O teste foi efectuado com os servidores X/Windows (PC e Terminal) configurados na resolução 1024x768x8bit (máximo denominador comum), recorrendo à aplicação *x11perf*. Cada barra representa o número médio, em iterações por segundo, calculado após 30 repetições consecutivas da *benchmark**.

**Da esquerda para a direita, cada entrada do gráfico corresponde aos seguintes atributos de teste da bateria:*

Dot	*1x1 rectangle*	*10x10 rectangle*	*100x100 rectangle*	*500x500 rectangle*
1x1 stippled rectangle (8x8 stipple)	*10x10 stippled rectangle (8x8 stipple)*	*100x100 stippled rectangle (8x8 stipple)*	*500x500 stippled rectangle (8x8 stipple)*	*1x1 opaque stippled rectangle (8x8 stipple)*
10x10 opaque stippled rectangle (8x8 stipple)	*100x100 opaque stippled rectangle (8x8 stipple)*	*500x500 opaque stippled rectangle (8x8 stipple)*	*1x1 tiled rectangle (4x4 tile)*	*10x10 tiled rectangle (4x4 tile)*
100x100 tiled rectangle (4x4 tile)	*500x500 tiled rectangle (4x4 tile)*	*1x1 stippled rectangle (17x15 stipple)*	*10x10 stippled rectangle (17x15 stipple)*	*100x100 stippled rectangle (17x15 stipple)*
500x500 stippled rectangle (17x15 stipple)	*1x1 opaque stippled rectangle (17x15 stipple)*	*10x10 opaque stippled rectangle (17x15 stipple)*	*100x100 opaque stippled rectangle (17x15 stipple)*	*500x500 opaque stippled rectangle (17x15 stipple)*

Tendo em conta estas circunstâncias, o Projecto OpenDMS oferece suporte explícito para a criação de *thin clients* a partir de componentes *of-the-shelf*. Esse suporte existe a dois níveis. Por um lado, pelo recurso ao PXE para carregar directamente via rede os sistemas operativos dos *thin clients* (arquivados nos servidores). Por outro lado, pela disponibilização de um conjunto de módulos (clientes gráficos, sistemas de ficheiros remotos, serviços de rede, etc.) que, em conjunto, permitem construir rapidamente *thin clients* à medida para "*hardware* PC". Estes módulos, maioritariamente adaptados do *Linux Terminal Server Project* (LTSP [LTSP]), permitem quatro modos distintos de operação:

- **Sistema *diskless* com interface gráfica.** Este sistema dispensa a existência de meios de armazenamento de massa locais, arrancando via rede, a partir de um *cluster* de servidores, acedendo a sistemas de ficheiros remotos por NFS, e utilizando uma pequena *ramdrive* local para *cache* de ficheiros frequentemente usados. O recurso a um servidor X local fornece suporte para aplicações executadas remotamente (por exemplo num servidor do *cluster* de apoio) e para os clientes WTS/RDP ou Citrix (que permitem aceder a servidores *Microsoft Windows Server* com serviços de terminal remoto). A possibilidade acrescida do acesso a áudio local, portas série e paralela e armazenamento amovível (usando o protocolo NBD) torna esta configuração extremamente flexível e adaptável, por exemplo, a quiosques multimédia. Adicionalmente, existe um modo especial de *helpdesk* suportado por esta configuração em que um PC normal pode arrancar directamente em modo gráfico, mostrando uma janela de um *browser* apontada para a página de requisição de intervenção pelo serviço de manutenção. Isto permite, por exemplo, usar o próprio PC danificado para solicitar apoio de *helpdesk*, mesmo com discos ou sistemas de ficheiros inoperacionais.

- **Network Computer.** Este modo corresponde a uma extensão natural do modo de operação *thin client*, possuindo o sistema meios de armazenamento de massa locais fixos para armazenar dados e programas frequentemente utilizados.

- **Network Appliance.** Neste modo o sistema arranca via rede com uma imagem contendo uma mini-distribuição de Linux especialmente desenvolvida para transformar o PC numa *network appliance*. Este modo adequa-se, por exemplo, à criação de servidores de impressão, dispositivos NAS (*Network Attached Storage*) ou combinações *router/firewall* baseados em hardware "PC".

- **Modo de Recuperação.** Este modo é pensado para permitir a recuperação remota ou localmente iniciada de um PC com um sistema de ficheiros seriamente danificado. Por iniciativa do servidor de gestão ou do utilizador do *desktop*, o PC arranca neste modo para descarregar uma imagem do seu sistema de ficheiros previamente salvaguardada e arquivada num servidor do *cluster* de apoio. Este modo de operação funciona com o auxílio de um conjunto de ferramentas licenciadas sob GPL (como o utilitário *parted* para manipulação de partições em disco) e executadas por *scripts* sobre uma mini--distribuição Linux desenvolvida para o efeito.

O aspecto mais curioso dos *thin clients* baseados na plataforma PC reside no facto de estes não necessitarem de *hardware* sofisticado. De facto, é possível construir, a partir de componentes elementares (por exemplo um processador Pentium 90MHz, 32MByte de memória, *bus* PCI, e placa de rede com suporte PXE), *thin clients* capazes de fornecer uma plataforma mais que apropriada para trabalho de escritório – tornando-se assim possível alargar a vida útil de uma configuração desactualizada que de outra forma seria posta fora de serviço, adiando o seu EOL (*End Of Life*) com uma reconversão para *thin client* com baixos requisitos de manutenção.

3.1.5 A Solução OpenDMS em Funcionamento

A Figura 3.6 apresenta um possível cenário de integração do OpenDMS. Neste cenário existem estações de trabalho Linux e Windows com uma configuração tradicional (sistema operativo armazenado localmente), PCs com personalidade múltipla capazes de comutar entre

os modos de operação *fat* e *thin client* de acordo com as circunstâncias, *thin clients diskless* e uma *network appliance* (um pequeno *router/firewall*). Existe um servidor OpenDMS dedicado à gestão de *desktops* e outro para suporte dos *thin clients*.

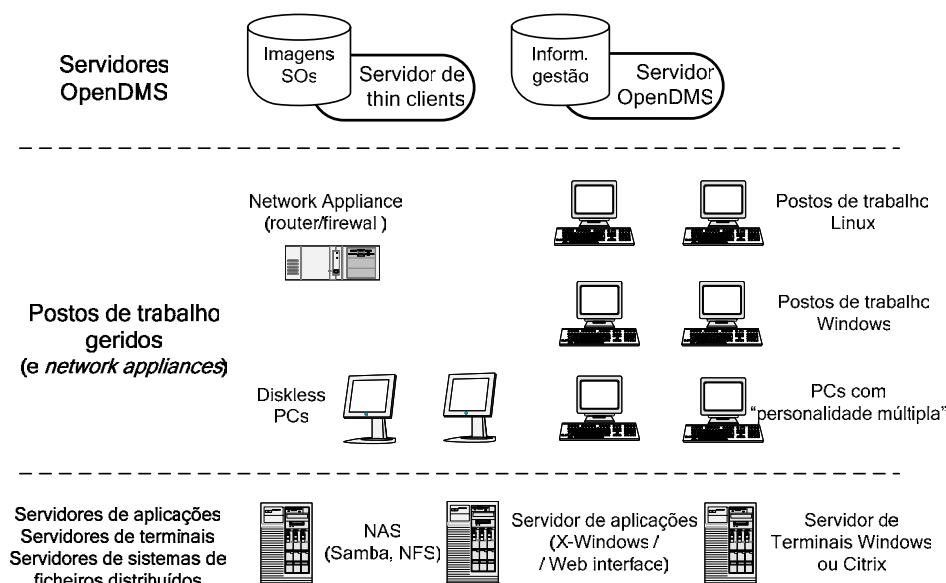



Figura 3.6 – Cenário de Gestão com Recurso à Plataforma OpenDMS

Para PCs com sistema operativo instalado localmente e configuração fixa (*desktops* Windows e Linux), o agente PreOS é utilizado principalmente para diagnóstico de hardware e verificação de integridade. Em caso de falha crítica do hardware os PCs podem ser bloqueados enquanto se aguarda uma intervenção *in situ*. Se a ocorrência se limitar ao nível de armazenamento de massa, sistema operativo ou sistema de ficheiros, o PC pode arrancar num modo especial onde uma mini-distribuição de Linux é descarregada via rede (a partir do servidor de *thin clients*) com meios para arrancar em modo gráfico com um *browser* apontado para a página de um serviço de *trouble tickets*. Alternativamente, se o dano apenas for a nível lógico (sistema operativo ou de ficheiros), existe a possibilidade de descarregar uma imagem de uma mini-distribuição de Linux preparada para recuperar uma cópia de segurança das partições (esquema e conteúdo) previamente salvaguardada e armazenada no servidor OpenDMS. Após carga do sistema operativo local, o agente *runtime* irá garantir a monitorização e gestão dos postos de trabalho.

Graças à flexibilidade oferecida pela solução OpenDMS, é possível ter postos de trabalho com meios de armazenamento de massa locais e "personalidade múltipla", capazes de operar como PCs tradicionais, como NetPCs ou como *thin clients diskless*.

No cenário apresentado existe também uma *network appliance* com funcionamento bastante simples: cada vez que é ligada, esta descarrega via rede uma imagem contendo um ambiente de operação especificamente desenvolvido para o efeito. Não existem sistemas de ficheiros locais para manter e as falhas de *hardware* mais críticas são facilmente solúveis: basta pegar num outro PC (ou substituir os componentes avariados), mudar a identificação da *appliance* nos servidores OpenDMS e ligar o novo PC, que inicia a sua operação mal o software seja descarregado.

3.2 Balanço do Projecto OpenDMS

Três anos depois da conclusão do projecto OpenDMS, não deixa de ser supreendente constatar a validade e actualidade dos problemas que constituiram o *leitmotiv* para o arranque da iniciativa. Continua a proliferação de *fat clients* nos postos de trabalho e, apesar da crescente sofisticação das ferramentas de gestão distribuída, as equipas de gestão parecem estar condenadas, *ad aeternum*, a lidar com a mesma classe de problemas.

O OpenDMS foi um projecto académico desenvolvido como balão de ensaio tecnológico para antecipar, explorar e potencializar algumas tendências e tecnologias inovadoras à época. Nessa óptica – ainda que o actual cenário continue insatisfatório – o balanço é positivo: constata-se que muitas das tendências antecipadas pelo OpenDMS se confirmaram, assim como algumas das soluções preconizadas.

Confirmando a previsão formulada há quatro anos atrás, segundo a qual a emergência de soluções mais adequadas para o problema da gestão distribuída de parque informático seria uma inevitabilidade, a indústria disponibiliza actualmente o tipo de recursos que originalmente eram preconizados no esboço da arquitectura OpenDMS, com será discutido nas próximas subsecções.

3.2.1 Agentes de Gestão *runtime*

Ainda que os agentes de gestão *runtime* instalados sobre o sistema operativo já fossem correntes há quatro anos atrás, verificou-se uma evolução notória a este nível, em especial pela perda de relevância de agentes proprietários e desenhados especificamente para uma aplicação de gestão e pela emergência de normais abertas para interfaces de gestão remota.

Presentemente, e falando do caso específico da plataforma Windows – a mais difundida actualmente – deve salientar-se o esforço feito pela Microsoft para a dotar de meios abertos e adequados para a gestão de *desktops*, destacando-se:

- **a inclusão de agentes SNMP (Simple Network Management Protocol) [RFC 1157] e WMI (*Windows Management Instrumentation*) [Microsoft 2005].** Por definição, as versões mais recentes dos sistemas operativos da família Windows incluem um agente WMI integrado e funcional, podendo adicionalmente instalar-se, em opção, um agente SNMP. A especificação WMI não é nada mais do que uma implementação da arquitectura WBEM que recorre ao CIM (sendo portanto baseada em normas abertas) para descrição do modelo de dados e que vem dar à plataforma Windows os mecanismos de inventariado, diagnóstico e controlo remoto de que necessita. Podemos por exemplo com um simples *script* escrito em *vbscript* [Microsoft 2005a] ou recorrendo à WMIC[4] para fazer uma consulta aos PCs de uma rede com recurso à WQL (*WMI Query Language*) para procurar quais os que correspondem a um determinado atributo. Este exemplo pode ser útil em situações em que se planeiam *upgrades* ou instalações colectivas de aplicações e se necessita de saber quais os postos de trabalho que possuem determinadas características de CPU, disco e memória para suportar o processo.

  Esta situação contrasta com o que sucedia há quatro anos atrás – com pouco suporte do lado do sistema operativo, qualquer agente com pretensões de adequação tinha de recorrer a subterfúgios para conseguir obter informações de inventariado, *runtime* do sistema e afins. A visão OpenDMS baseava-se no pressuposto de que um agente não deveria ser uma entidade pesada (em alguns casos mais pesada que a maioria das aplicações) e que deveria ser o ambiente do sistema (sistema operativo e *firmware*) a fornecer os *building-blocks* para os agentes. Se bem que à data do início do projecto já existissem recursos como o DMI/SMBIOS, só com o aparecimento de extensões como o WMI, embebidas no sistema operativo, passaram a existir mecanismos confiáveis

[4] *WMI Command-Line Tool:* ferramenta de linha de comando incluída na plataforma Windows (XP ou superior) com o objectivo de simplificar o uso e acesso à WMI

para fazer a gestão *online* dos postos de trabalho baseados em Windows, melhor integrados com os *standards* de baixo nível existentes.

- **a criação de mecanismos de gestão distribuída centralizados.** Estes mecanismos, baseados na noção de *policy management*, permitem gerir grupos de PCs e utilizadores e controlam o acesso destes últimos a vários aspectos e funções dos postos de trabalho, desde a restrição de instalação de aplicações até ao estabelecimento de limitações específicas do seu uso, permitindo ainda efectuar o *push* de software para as máquinas de um determinado grupo. Quando devidamente utilizados, permitem salvaguardar a integridade dos postos de trabalho, limitando um dos principais motivos da má reputação angariada pelo PC: o excesso de liberdade dos utilizadores.

  Neste aspecto, o projecto OpenDMS propunha uma solução mais radical, baseada não apenas no reforço das características de controlo e recuperação dos PCs ao nível do sistema operativo mas também na adopção de *thin clients* de baixa manutenção em substituição dos *fat clients*, para as tarefas mais usuais (aplicações do tipo *office*, CRM, web *browsing*, E-mail, entre outras).

- **Software Update Services (SUS) [Microsoft 2004] e Windows Update Services (WUS) [Microsoft 2004a].** Este serviço permite automatizar a distribuição de actualizações dos PCs Windows de uma rede, com enormes benefícios em termos de uso de largura de banda. Ao configurar os PCs cliente para recorrerem a um servidor interno à rede – e, portanto, mais próximo – para as actualizações, em vez de se conectarem ao exterior, o administrador consegue ter maior controlo sobre as alterações feitas nos postos de trabalho (mandam as boas prácticas testar todas as actualizações num sistema cobaia, antes de as distribuir pelos postos de trabalho) e poupa largura de banda nas ligações ao exterior da rede local. A nova geração do serviço SUS, designada WUS, permite ainda efectuar actualizações de software aplicacional, contemplando todos os produtos da linha Office e Server da Microsoft, como é o caso do Exchange, ISA Server, SQL Server, entre outros.

  A necessidade de mecanismos de actualização de software é, em grande medida, dependente de cada sistema operativo, pelo que nunca se considerou que o projecto OpenDMS intervisse neste aspecto.

- **Remote Assistance/RDP.** No Windows XP encontra-se incluído um servidor RDP com capacidade para uma ligação remota, mutuamente exclusiva com o uso local do PC. Contudo, se for utilizado o mecanismo de *remote assistance*, é possível ao utilizador de um PC enviar um pedido de ajuda via correio electrónico (desde que se utilize uma conta *MAPI-compliant*) ou *MSN Messenger* a alguém que se poderá conectar ao PC e visualizar em tempo real o ecrã remoto e, caso lhe seja concedida permissão, aceder ao controlo do rato e teclado para auxiliar na resolução do problema.

  O projecto OpenDMS já suportava este tipo de mecanismos, recorrendo para o efeito ao VNC, de modo a suportar o número de plataformas o mais amplo possível com um só cliente e não apenas sistemas Windows/RDP.

Ainda no respeitante a este tipo de mecanismos, surgiram entretanto algumas soluções multiplataforma, como é o caso das propostas existentes no catálogo de produtos da Altiris [Altiris], contemplando os aspectos da gestão de *desktops*, servidores/infraestrutura, inventário e segurança. Estas soluções, apesar da sua versatilidade e funcionalidade, pouco acrescentam de verdadeiramente novo ao conceito, sofrendo das fragilidades inerentes aos mecanismos contextuais a sistemas operativos e acarretando custos extra de formação e licenciamento.

3.2.2 Mecanismos PreOS ou OS-*absent*

Com a emergência das BIOS UEFI (*Unified Extensible Firmware Interface* [UEFI]) de nova geração, passaram a estar disponíveis agentes PreOS estáticos, executados em contexto prévio

à carga de sistema operativo, embebidos no *firmware*, possuindo uma interface Web e permitindo operações de diagnóstico o sistema, acesso a bases de conhecimento ou comunicação remota de falhas (iniciada de forma manual ou automática). O projecto OpenDMS propunha agentes PreOS dinâmicos, descarregados no estágio prévio à carga do sistema operativo, com o auxílio de uma *boot* ROM PXE, com um propósito essencialmente idêntico ao dos agentes estáticos UEFI mas dispondo de maior flexibilidade. Os agentes estáticos das BIOS UEFI necessitam de qualquer forma de conectividade de rede para efectuar a maioria das suas funções e, estando embebidos no *firmware*, a sua actualização é mais complexa que a dos agentes PXE dinâmicos (descarregados via rede no estágio PreOS), que podem ser actualizados através de um simples *update* ao servidor de descarga/manutenção e por esse meio afectar disseminados para todo o parque gerido. Sendo ambos basicamente inúteis sem conectividade de rede, a vantagem pende claramente a favor do modelo de agente dinâmico baseado nas extensões PXE (cuja inclusão está aliás prevista na norma UEFI).

Também deverá ser mencionado o facto de, nos sistemas Windows, os serviços de instalação remota – designados por *Remote Installation Service* (RIS) no Windows 2000 Server [Microsoft 1999a] e *Windows Deployment Services* (WDS) no ambiente Windows 2003 Server [Myers 2005] – suportarem o recurso a *boot* ROMs PXE para permitir a instalação de PCs sem necessidade de recorrer a meios de armazenamento amovíveis. Existe mesmo uma versão do Windows, a *Preinstallation Environment* (PE) [Myers 2005] que é criada com recurso a um *toolkit* a partir de uma instância já instalada de Windows XP, 2000 ou 2003 e que pode executar a partir de um meio *read-only* (CD, DVD ROM), podendo ser carregada via PXE para teste ou instalação de PCs.

O projecto OpenDMS contempla estas funcionalidades de forma mais abrangente, devido ao facto de estar pensado com a neutralidade de plataforma em mente, permitindo a salvaguarda de instâncias de sistema operativo num servidor central, a partir do qual poderiam ser repostas mais tarde.

A DMTF desenvolveu também trabalho nesta área, apresentando o *Alert Standard Format* (ASF) [DMTF 2001b] com o propósito de propor uma especificação orientada para a gestão de máquinas num estado OS-*absent* (como um portátil em modo de hibernação ou um PC desligado). O suporte ASF deve ser embebido no hardware do PC, permitindo ao administrador, através de uma consola de gestão, controlar o sistema e ser notificado de ocorrências específicas, tais como problemas ambientais (temperatura, tensão de alimentação, ventilação), de processador, de firmware, de memória, falhas de arranque, entre outros. Entre as opções de controlo suportadas encontram-se a possibilidade de requisição das características ou estado, *ping*, *power-off* (incondicional ou não), *power-on* ou *reset* do sistema. Contudo, apenas com a versão 2.0 desta norma foram contemplados os aspectos de segurança, pelo que a grande maioria das anteriores implementações não possuía as opções de *power-on* e *power-off* remoto.

A Intel, por sua parte, desenvolveu a *Active Management Techology* (iAMT) [Bogowitz 2005], baseada em mecanismos de baixo nível implementados no *firmware* e *hardware* do sistema e que possui as seguintes características:

- capacidade de gestão *out-of-band* (OOB), independentemente de o PC estar ligado ou desligado e do estado da instância de sistema operativo.
- suporte para diagnóstico e recuperação remota de PCs.
- implementação da norma ASF.
- uso de memória não-volátil para criação de agentes persistentes, resistentes a intrusão (*service OS*) capazes de operar independentemente do facto de a máquina estar ligada, em operação, em *crash* ou apenas com problemas. Esta memória pode também ser utilizada para armazenar informação de gestão do sistema (de inventário, por exemplo) que pode ser acedida com o PC desligado.

- *tracking* de *hardware* e *software*.
- solução agnóstica em termos de plataformas suportadas graças a mecanismos de gestão *out-of-band* independentes de qualquer sistema operativo.

A maior parte destas funções estava contemplada no agente PreOS OpenDMS, mas deve reconhecer-se que a aproximação seguida pela Intel, modificando e concebendo *hardware* para conseguir os objectivos supracitados, no sentido de criar um ambiente de *embedded management* no PC apenas é possível a um fabricante com a sua dimensão. Isto não invalida a aproximação OpenDMS à questão, pois o iAMT só está implementado em *hardware* muito recente, não sendo ainda suportado na maioria dos componentes OEM existentes no mercado e não podendo ser de forma alguma "adicionado" a sistemas mais antigos. Deste modo, a solução OpenDMS constitui ainda uma alternativa satisfatória nos sistemas que não suportam *embedded management* (e que constituem a larga maioria do parque informático existente).

3.2.3 *Thin Clients*

Há quatro anos atrás vivia-se na fase de rescaldo da euforia dos *network computers*, netPCs e afins, pelo que os *thin clients* em geral gozavam de uma imagem francamente negativa. Porém, na perspectiva do gestor de sistemas, os *thin clients* constituiam a solução para muitos dos problemas gerados pela disseminação de PCs pelos postos de trabalho. Ainda que aparentemente um conceito promissor, o paradigma padecia de sérios problemas de concretização, como resultado de limitações tecnológicas e recurso a soluções técnicas inadequadas que, em última análise, o tinham colocado em cheque.

Na perspectiva OpenDMS existia um lugar pertencente por direito aos *thin clients*, por contraposição à proliferação de *fat clients* pelas organizações, frequentemente injustificada. Procurando identificar as lacunas da geração anterior de produtos comerciais, concebeu-se no OpenDMS um sistema com baixo custo de implementação, flexível, robusto e resiliente, baseado em normais abertas e difundidas, capaz de tolerar falhas nos servidores de apoio e continuar a operar. Aliado a um conjunto de *guidelines*, o *thin client* OpenDMS era proposto como um caminho para reconciliação do conceito com os gestores de sistemas.

Actualmente, constata-se que os grandes integradores de sistemas de *Tier-1* (HP, Dell, Fujitsu-Siemens, entre outros) possuem *thin clients* nos seus catálogos e que as empresas cujo *core business* se baseia apenas neste tipo de sistemas, como a WYSE, a NCD e a Igel, estão de boa saúde. Mais importante, a Sun Microsystems (um dos proponentes do conceito do *network computer*) não só nunca abandonou o conceito como adoptou o uso extensivo de *thin clients* SunRay [Sun 2004] nos próprios postos de trabalho da organização. Além disso, nos últimos tempos muitas organizações descobriram que a solução mais simples para a manutenção de muitas aplicações legadas passa pela adopção de *thin clients*, evitando o custo de adaptação e migração para outras plataformas.

Saliente-se apenas que os produtos comerciais actualmente disponíveis (*cfr. caixa*) se assemelham em muitas das suas características ao sistema-tipo proposto pelo projecto OpenDMS, com excepção de dois pormenores: continuam vulneráveis a falhas de comunicações e dos servidores, devido ao uso de protocolos *unicast*; e dependem de um ambiente de operação local e estático, embebido em *firmware*. Avaliando a primeira lacuna, e relativizando-a em função da situação presente, constata-se que esta já não constitui um factor crítico, pois a evolução nos níveis de fiabilidade nas infrastruturas de comunicações e serviços diminuiu a sua relevância de forma significativa. Em relação à segunda lacuna, e tendo presente que estes sistemas já são inerentemente dependentes da conectividade de rede (mesmo que o ambiente de operação esteja em *firmware* e o sistema arranque, este não é usável sem acesso aos servidores), o modelo de operação proposto pelo projecto OpenDMS mantém a sua pertinência: baseando-se numa imagem contendo o sistema operativo que é descarregada aquando do arranque, os *updates* tornam-se numa operação simples, bastando actualizar as imagens existentes nos servidores de gestão e manutenção para disseminar os *updates* por todo o parque gerido.

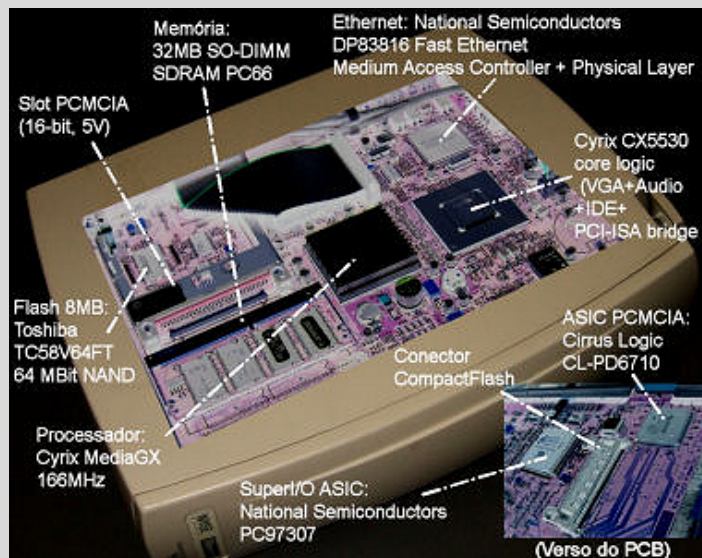
Arquitectura de um *thin client* moderno vista ao Raio-X
WYSE Winterm 3320SE

O *copyright* da *motherboard* deste *Winterm 3320SE* data de 2000, encontrando-se dotada de uma CPU x86 de proveniência Cyrix/National Semiconductors (actualmente o *brevet* é propriedade da AMD) - o MediaGX, cuja primeira geração remonta a 1997.

Desenvolvido tendo em vista o mercado dos PCs x86 de baixo custo, o MediaGX baseia-se numa arquitectura *x86* agregando num só integrado as funções da CPU, controlador VGA (do tipo UMA – *Unified Memory Architecture*, recorrendo à memória principal do sistema para o subsistema de vídeo) *interface* de memória e gestão de energia. Possui uma cache de 16Kb L1 estando ainda dotado de uma *pipeline* simples de 6 estágios e uma FPU.

O resto das funções de *interface* é efectuada pelo ASIC CX5530 que fornece uma *bridge* PCI-ISA, controlador IDE DMA e Audio. O terminal recorre ainda a um ASIC SuperI/O para poder dispor de duas portas série RS232C (para conectar um rato série, modems ou interface de *touchscreen*), uma porta paralela (para impressora) e duas portas PS/2 (para rato e teclado). Possui um slot PCMCIA que pode ser utilizado para estender a funcionalidade do terminal, permitindo-lhe operar em redes *wireless*, *token ring*, suportar um adaptador RDIS ou uma segunda *interface* de rede *ethernet.*

O *firmware* do sistema (sistema operativo Windows CE 2.x, 3.x ou CE .NET – inclui cliente *RDP*, *Telnet* com suporte de 15 emulações de terminal distintas, *Citrix* e *browser Internet Explorer 4.0*) encontra-se armazenado numa memória *flash* de 8MB, que pode ser estendida com o recurso ao conector *CompactFlash* existente no verso e que se encontra ligado ao *interface* IDE do *chipset*. Com 32MB de memória SDRAM (expansíveis até 64MB) possui a capacidade de suportar sessões concorrentes de *telnet*, *RDP* ou *Citrix* (em algumas versões mais recentes suporta SSH – o número de sessões varia conforme a configuração específica) com redireção do fluxo de dados dos periféricos e audio para o terminal.

Como a alimentação do *Winterm* é fornecida através de um bloco externo e o processador se encontra acoplado a um dissipador passivo não existem quaisquer peças móveis passíveis de falhar ou causar ruído.

3.2.4 Mecanismos de Diagnóstico, Recuperação e Resiliência

Um dos grandes problemas do PC moderno é a sua completa incapacidade de tomar conta de si próprio: sem o devido *babysitting*, pode tornar-se uma *prima donna* problemática e de manutenção dispendiosa. No modelo OpenDMS, a solução proposta passava por dois aspectos:

- estabilizar um conjunto de imagens de sistema para as plataformas geridas, que seriam utilizadas para instalações automáticas ou operações de recuperação com recurso a intervenção local mínima por parte das equipas de gestão.

- recorrer aos agentes PreOS e *runtime* para obter informação de estado da máquina para efeitos de diagnóstico, operações de manutenção preventiva e vigilância activa.

No capítulo da recuperação, podemos actualmente contatar que a maioria das soluções existentes evoluíram no mesmo sentido que o OpenDMS. Um exemplo disso mesmo é a iniciativa *Stable Image Platform Program* (SIPP) [Intel 2005], da Intel, cujo objectivo é desenvolver um programa de apoio à criação de "imagens estáveis" de instâncias de sistemas operativos e aplicações nos PCs, para facilitar o *deployment* automatizado e massificado de PCs (podendo este processo ser este levado a cabo com recurso a serviços como o RIS/WDS, que quando articulados com os procedimentos de *unnatended install* permitem levar a cabo instalações completas de PCs com intervenção mínima do administrador). Outra das funções destes serviços é a de permitir o *pre-staging* de PCs Windows, ou seja, após a instalação de um PC, criar uma instância de ambiente de operação (sistema operativo e aplicações) no servidor, feito a partir da máquina recém-instalada e que poderá ser utilizado para futuras instalações da mesma máquina ou de outras máquinas cujo hardware seja practicamente idêntico ao do sistema original.

Como já foi mencionado, a recente tecnologia iAMT da Intel possui também este tipo de características, mas não estão ainda disponíveis quaisquer plataformas de gestão capazes de tirar partido ela.

No capítulo do diagnóstico e resiliência, a baixo nível, merecem destaque as tecnologias ASF e iAMT, que permitem efectuar o diagnóstico remoto de PCs e que podem ser articuladas com os mecanismos de alto nível já existentes (WMI, agentes de gestão *online*) para oferecer informações completas de estado. Estão também disponíveis os agentes estáticos (U)EFI PreOS e, pela primeira vez, sob a forma da iAMT, está disponível uma tecnologia que prevê mecanismos robustos de *embedded management* capazes de funcionar independentemente do estado do *hardware* para diagnosticar e reparar um PC.

A alto nível, os agentes *online* (caso dos agentes WMI e SNMP na plataforma Windows) permitem efectuar remotamente o diagnóstico e vigilância dos sistemas no contexto do ambiente de operação do sistema operativo.

3.2.5 Interoperabilidade entre Plataformas

Outro dos calcanhares de Aquiles das soluções de gestão de parque informático existentes há quatro anos atrás era derivado da sua natureza proprietária, comprometendo muitas vezes de forma irreversível as opções tomadas a determinados vendedores e/ou soluções fechadas. Presentemente a adopção crescente de normas abertas e flexíveis (WBEM, SNMP, entre outras) mostra que a indústria já percorreu um caminho bastante significativo.

A título de curiosidade, não deixa de ser interessante que o projecto OpenDMS considerasse o recurso ao agente *online* para fazer a integração com a WMI – devido a uma lacuna na proposta do DMTF para o CIM, não se especificavam mecanismos para a troca de informação. Como resultado dessa lacuna, a Microsoft decidiu implementar um protocolo para troca de informação CIM sobre COM/DCOM (*Component Object Model/Distributed Component Object Model*) [Microsoft 1996], tornando-o incompatível com sistemas não-Windows. Apenas recentemente o DMTF se debruçou sobre a questão da interoperabilidade, tendo proposto o protocolo CIM-XML [DMTF 2002], contemplando um formato DTD (*Document Type Definition*) [DMTF 2002a] que mapeia os atributos CIM num documento XML (*eXtensible Markup Language*) [W3C 2004] e um mecanismo de transporte sobre HTTP, designado *CIM Operations over HTTP* [DMTF 2004].

A opção do projecto OpenDMS por um agente próprio contrasta com a via tomada por algumas soluções mais recentes, que recorrem a camadas de translação WMI-SNMP, com MIBs que incluem contadores de desempenho contemplando desde os *core four* (rede, CPU, memória, disco) até milhares de indicadores que podem transcender o próprio sistema operativo e incluir informação de aplicações ou serviços.

3.2.6 Conclusão

Apesar de toda a evolução e dos progressos levados a cabo no mundo *wintel*, a tarefa de gestão de *desktops* continua uma tarefa complicada e problemática.

Os PCs não deixaram de ser máquinas com fragilidades, muitas vezes sobredimensionadas para as tarefas que desempenham e cuja gestão continua a ser dispendiosa. Para pôr em prática a maioria dos progressos recentemente alcançados em termos de gestão seria necessário, em alguns casos, substituir mais de 90% dos PCs em operação (caso do suporte iAMT) e dispender um montante considerável em formação e ferramentas para poder beneficiar das novas tecnologias e avanços. Atente-se ainda que, apesar da adopção de normas abertas, persistem problemas de interoperabilidade entre plataformas distintas que, em última instância, condicionam de forma crítica a escalabilidade das soluções existentes, ao ponto de nalguns casos fomentarem *overheads* nas equipas de suporte e gestão de TIs na ordem dos 50% [Busch 2004]. A visão da gestão remota multiplataforma ainda é uma utopia.

Os problemas de inventário persistem – apesar das soluções existentes, calcula-se que a maioria das organizações não seja capaz de localizar 20 a 25% (ou mesmo 33%, em casos extremos) dos seus *assets* [Busch 2004], não sabendo se estes foram vendidos, roubados ou descontinuados de serviço. Isto leva a situações de má gestão de compras, com consequente impacto no TCO da infraestrutura de TIs.

Para agravar ainda mais esta situação, não pode deixar de se considerar o surgimento de novas tecnologias e tendências que irão inevitavelmente originar novos problemas para as equipas de suporte. Desde a emergência das redes 802.11 até ao crescendo de importância do VoIP (*Voice over IP*), podemos vislumbrar um terreno fértil onde já se faz sentir novamente a necessidade de soluções, ideias e, acima de tudo, formas diferentes de "fazer as coisas".

3.3 Um Novo Paradigma: *Balanced Computing Platform*

3.3.1 Motivação

À semelhança do que sucedeu no passado, encontramo-nos novamente na emergência de um novo paradigma: o da ubiquidade e do *pervasive computing*. À medida que os meios informáticos se tornam *commodities*, os utilizadores começam a exigir que estes se tornem mais "transparentes" na forma de utilizar e interagir. A tal comportamento não são alheios o súbito aumento da capacidade de processamento por Watt, a difusão massificada das redes *wireless* (com a consequente redefinição do conceito da computação móvel) e o crescimento que se tem vindo a observar na difusão do VoIP. Começa a tornar-se apreensível a iminência de uma mudança de fundo.

A mobilidade acrescida dos utilizadores tem como principal consequência a concretização de um dos piores pesadelos dos administradores de sistemas: se anteriormente existia um sistema completo em cada posto de trabalho, com os respectivos periféricos e estado local (instância de sistema operativo com programas e dados locais), havia pelo menos, na maioria dos casos, a garantia da relativa imobilidade desse mesmo posto. Com a crescente mobilidade, o conceito de "posto de trabalho" passa a ser algo muito relativo. Essa flexibilidade acrescida é sem dúvida muito conveniente para os utilizadores mas constitui uma nova fonte de problemas para as equipas de gestão.

Adicionalmente, há outro tipo de mobilidade que começa a ganhar adeptos entre algumas organizações. Tornando os cubículos/postos de trabalho num recurso, e não num lugar garantido e fixo, promove-se uma política de flexibilidade em que o trabalhador ocupa um determinado posto durante o horário de expediente que deverá ser libertado se este tiver de se ausentar por um período superior a um valor estipulado, podendo ocupar outro local aquando do seu regresso se o anterior já estiver ocupado. Para viabilizar este modelo de uso dos recursos físicos exige-se que o utilizador tenha acesso ao seu ambiente de trabalho e dados a partir de para onde quer que se desloque, de forma transparente.

A reacção da indústria das TIs às novas tendências

Um pouco por todo o lado constata-se o surgimento de um novo conjunto de iniciativas que procuram, de forma análoga ao sucedido no passado, redefinir o PC para tirar partido dos avanços tecnológicos mais recentes – é o caso das especificações *MicroBTX e picoBTX* [Intel 2005c], da especificação *Nano-ITX* e respectivos protótipos [VIA 2004, VIA 2004a] da VIA e das iniciativas *Desktop Platform Vision 2006* [Intel 2005a] e *Platform 2015* [Intel 2005b], da Intel. No caso da *Desktop Platftorm Vision 2005/2006* e da *Platform 2015,* não se trata apenas de um *remake* do PC clássico em versão miniaturizada, *power-efficient* e/ou esteticamente aprazível, mas de uma visão mais ampla que contempla pela primeira vez mecanismos de gestão contextualizados (ambas as propostas contemplam protótipos destinados especificamente a ambientes SOHO e empresariais), como é o caso do iAMT.
Infelizmente, os protótipos que se conhecem (na figura, o protótipo *Averill* da Intel), derivados das propostas existentes têm algo em comum – apesar da adopção de um vasto conjunto de tecnologias (são de facto o que se apelida de *technology enablers* ou *technology vehicles*) e funcionalidades, todos se apresentam sob a forma de PCs melhorados e sofisticados, mas que não deixam de ser PCs.
Esta atitude conservadora é resultante da tradicional resistência da indústria em geral aos novos paradigmas de uso – desde a passagem dos sistemas de grande porte para o modelo cliente-servidor até ao reconhecimento do potencial de conceitos mais recentes, como foi o caso do *Media Center*, a evolução surgiu mais com mais frequência como resultado da pressão dos utilizadores e pioneiros do que dos *technology trendsetters*. De facto, quando a proliferação de soluções ou alternativas de natureza mais ou menos espontânea (caso do próprio computador pessoal), muitas vezes de cariz DIY (*Do-It-Yourself*) ganha forma e força nas análises SWOT ou quando o ciclo de vida de um produto há muito que deixou de estar na fase das *vacas gordas* é que surgem as vontades para encetar processos de evolução e mudança.

Adicionalmente, a difusão do VoIP em ambientes domésticos e empresariais traz novas exigências à própria infraestrutura de suporte em termos da qualidade de serviço prestado e acessibilidade. Inevitavelmente, alguns utilizadores começam a exigir ter acesso, na sua nova plataforma móvel, a tudo aquilo a que tinham direito no seu antigo posto de trabalho estático.

A palavra-chave nesta nova problemática é “plasticidade”. Como suportar uma massa de utilizadores sujeitos a um regime de mobilidade total ou parcial que exigem acesso a todos os recursos e serviços a que sempre estiveram habituados sem comprometer a segurança, transparência de acesso e mantendo nos postos móveis as características de manutenência que eram possíveis nos desktops fixos? O problema começa na própria visão da infraestrutura de rede: é necessário começar a abordar a questão da gestão distribuída de uma forma dinâmica, mais flexível e, acima de tudo, mais eficaz.

3.3.2 *Balanced Computing Platform*

O problema que se coloca com o novo paradigma da mobilidade pouco tem de novo e pode resumir-se a uma questão: ***como gerir de forma eficiente os meios sem limitar a liberdade dos utilizadores?***

Sejam quais forem as tecnologias a que se faça recurso, a solução para esta questão passará na maior parte das vezes por uma tentativa de tornar o posto de trabalho uma entidade idempotente, com estado local mínimo ou, idealmente, inexistente. No PC clássico, tal passa pela estabilização das configurações (tendo em vista procedimentos de reinstalação e recuperação o mais rápidos possível em caso de necessidade) e pelo controlo apertado do acesso aos meios de armazenamento de dados locais de natureza inamovível, promovendo em alternativa o recurso a meios de armazenamento centralizados disponíveis via rede. Um observador mais atento não poderá deixar de reparar que estes requisitos assentam perfeitamente no conceito de *thin client*. Foi precisamente por isso que estes foram “recuperados”, no projecto OpenDMS, como alternativa ao PC clássico para situações onde os utilizadores desenvolvessem trabalho típico de escritório (processador de texto, folha de cálculo, acesso *web*, correio electrónico, *front-ends* para aplicações cliente-servidor e afins). A ideia subjacente consiste em conceder PCs completos aos utilizadores que deles necessitem (*power users*) e disponibilizar *thin clients* aos restantes.

Porém, com a difusão de ferramentas de trabalho como a videoconferência e o VoIP, e com a crescente mobilidade dos utilizadores, constatam-se as limitações associadas aos *thin clients* neste novo contexto. Ainda que seja possível "adaptar" o conceito de *thin client* (o que no caso da interpretação OpenDMS seria relativamente simples, por esta estar baseada num paradigma híbrido *server/network-centric*) para lidar com as necessidades do VoIP e mesmo, eventualmente, de acesso a videoconferência, este não deixa de funcionar na maioria das vezes como um *display* remoto cujo desempenho começa a ser de todo incompatível até com a simples navegação na Web (graças à proliferação dos conteúdos multimédia dinâmicos e interactivos). Este problema de desempenho deriva do facto de não se recorrer à capacidade de processamento local destes *thin clients* para executar aplicações (esta é utilizada exclusivamente para o sistema operativo e para os clientes de protocolos de conexão remota), por estarem limitados à função de terminal gráfico remoto conectado a um servidor. É certo que alguns *thin clients* incluem no *firmware* um *browser* e uma JVM, à semelhança dos *Network Computers,* permitindo assim a execução local de aplicações, desde que escritas e/ou portadas para a linguagem *Java*, mas o desempenho e flexibilidade destes sistemas são, de modo geral, tão limitados que estas funcionalidades são sistematicamente postas de parte.

Posto isto, há quem defenda um regresso aos *fat clients* em cada posto de trabalho ou a adopção de uma nova geração de *thin clients* híbridos (como os apresentados recentemente pela *Igel* [Igel 2005]) capazes de resolver algumas das limitações, o que para alguns casos será apenas o adiar de uma inevitabilidade.

Há quatro anos atrás, fazia sentido propor o compromisso de fornecer sistemas mais limitados aos utilizadores, passando a complexidade para o lado da infraestrutura de comunicações e servidores e beneficiando assim das vantagens do paradigma centralizado clássico. Presentemente o modelo centralizado então proposto está a aproximar-se dos seus limites, sem que o PC constitua uma real alternativa. A solução passa assim por encontrar algo que permitia beneficiar do melhor dos dois mundos, aliando a facilidade de gestão do *thin client* com o desempenho do PC (Figura 3.7).

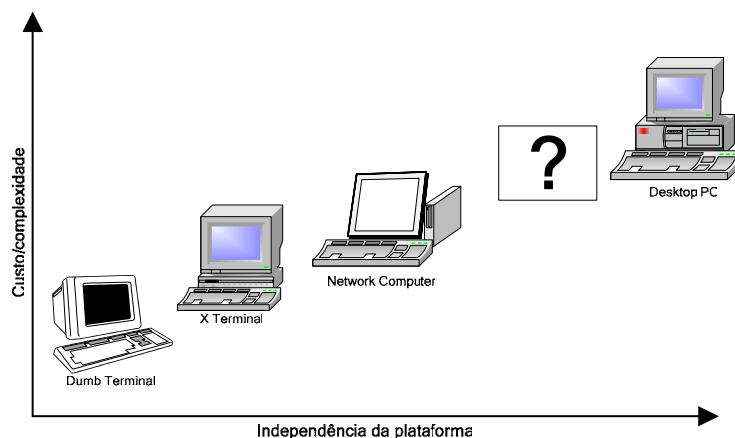



Figura 3.7 – Janela de Oportunidade para um Novo Paradigma

A Figura 3.8 ilustra de outra forma esta questão, propondo um modelo alternativo, que passaremos a designar por *balanced computing platform*. Este modelo permite aliviar a infraestrutura de servidores, por contraposição ao modelo *server-centric*, fazendo uso das capacidades de processamento locais e beneficiando assim da possibilidade de poder executar um leque de aplicações mais vasto e com requisitos de desempenho superiores. Adoptando este paradigma computacional por forma a privilegiar o recurso a meios de armazenamento remoto, acessíveis via rede, e limitando o acesso local a dados apenas a dispositivos amovíveis, é possível tirar partido do melhor que os *thin* e os *fat clients* têm para oferecer, sem sofrer com as limitações de cada sistema.

	Server-centric Model	Network-centric Model	**Balanced Computing Model**	Connected PC Model
Localização dos dados	Servidor	Servidor	**Clientes e/ou servidores**	Cliente e servidor de ficheiros
Localização das aplicações e contexto de execução	Servidor	Armazenadas no servidor, descarregadas para o cliente para execução	**Cliente e/ou servidor**	Cliente
Aplicações típicas	Aplicações centralizadas *business-oriented*, aplicações de consulta e/ou *browsing*	Aplicações *Windows* descarregadas aplicações,*Applets Java ou plugins ActiveX*	**Todo o tipo**	Aplicações tipo *Office*, componente cliente de aplicações *multi-tiered cliente-servidor*
Utilizador-tipo a que mais se adequa	*Light,* utilizador de aplicações do tipo mono-tarefa transaccional (ex: operadores de um *call-center* com aplicação CRM)	Funcionário Administrativo, tarefas de nível médio de complexidade (pequenos documentos, e-mail, *messaging*, *browsing* numa *Intranet*)	**Todos**	Avançado, *knowledge user* (documentos complexos, análise, processamento e criação de conhecimento, acesso a *Internet* e *Intranet*)
Implementações possíveis	PCs, network computers, X terminals	PCs, network computers	**(?)**	PCs

Figura 3.8 – *Balanced Computing Platform*, comparação com outros paradigmas

Pode mesmo afirmar-se, em tom de síntese, que o recurso a este paradigma tem como principal benefício a consolidação implícita de recursos, quer seja em termos de capacidade de processamento ou de armazenamento de dados. A explicação é simples:

- ao permitir equilibrar o uso das capacidades de processamento de forma mais efectiva, entre a infraestrutura de servidores e os clientes (cada cliente pode operar em modo *thin client*, *web client* ou utilizando aplicações locais) conseguem-se ganhos de considerável magnitude, aliviando o esforço de dimensionamento da estrutura de TIs para dar resposta à necessidades de escalabilidade (Figura 3.9). É preciso não esquecer que uma estrutura de servidores deve ser dimensionada para os propósitos a que se destina, com base em critérios de predictabilidade que muitas vezes têm de ser estimados em excesso (e quando se atinge o limite estimado pelas previsões originais entra-se numa espiral de upgrades de disco, memória e CPU em nome da escalabilidade que acaba por determinar a substituição do servidor). Ao permitir equilibrar de forma implicíta as capacidades de processamento de ambos os lados, a *Balanced Computing Platform* vem dar uma resposta a esta situação.

- ao privilegiar o recurso a meios remotos de armazenamento de dados, facilita-se a sua consolidação, pondo travão à sua proliferação e dispersão. São comuns as situações em que uma determinada máquina não dispõe de espaço em disco enquanto o PC ao lado possui um disco com 200GByte a 20% de ocupação. Como não é possível simplesmente "mover" um byte livre que seja do sistema com disponibilidade de espaço para ceder ao que dele necessita, a solução passa muitas vezes por um *upgrade* de disco, limpeza de sistema parcial/total ou, em situações extremas, a substituição do desktop (Figura 3.10).

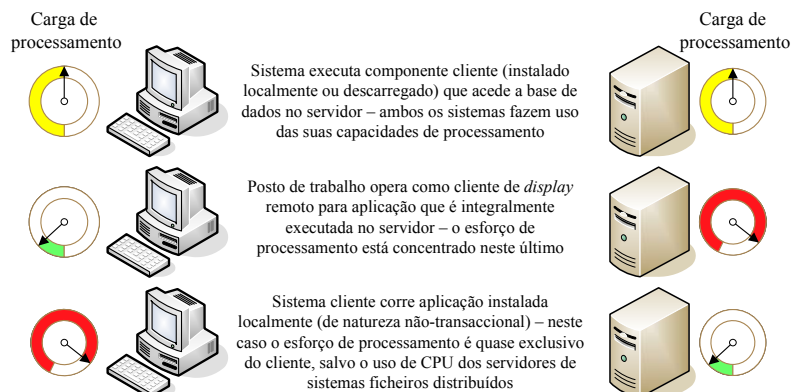



Figura 3.9 – *Balanced Computing Platform*: Gestão da Carga de Processamento

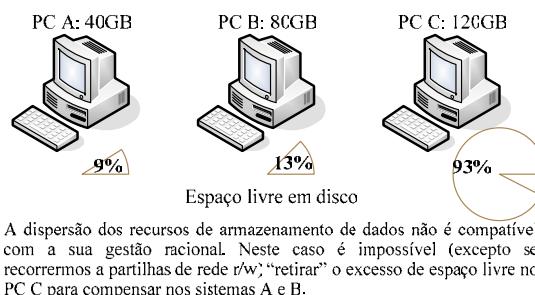



Figura 3.10 – *Balanced Computing Platform*: Gestão da Capacidade de Armazenamento

3.4 Projectos IC^3 e DOMUS

3.4.1 Projecto IC^3: *Integrated Communications and Computing Concept*

Em si mesmo o conceito de *Balanced Computing Platform* oferece apenas parte da resposta para os desafios colocados à gestão de *desktops* nos dias que correm. Para encontar o ponto de equilibrio entre os varios factores envolvidos na equação (Figura 3.11) é necessário integrar e validar o conceito de forma detalhada e integrada numa arquitectura concreta.

Figura 3.11 – Factores a Considerar no Projecto IC^3

Assim sendo, e tendo em conta uma perspectiva de conjunto, identificou-se um conjunto de características a incorporar numa interpretação concreta do conceito de *Balanced Computing Platform* – o projecto IC^3 – aqui apresentadas na forma de uma lista de requisitos funcionais:

- **Idempotência da plataforma**. O sistema deve ser desprovido de estado local tanto quanto possível, de modo a facilitar a sua manutenção e substituição em caso de falha.
- **Transparência no acesso a recursos e suporte para mobilidade**. A plataforma proposta deve viabilizar de modo eficiente a possibilidade de *roaming*, por forma a que os utilizadores possam aceder ao seu ambiente de trabalho e dados a partir de qualquer posto onde se autentiquem – permitindo assm uma grande flexibilidade no uso do espaço nos locais de trabalho.
- **Eficiência da relação MIPS/Watt**. A plataforma de referência IC^3 deverá estar concebida em torno de uma arquitectura com uma boa relação desempenho/consumo energético, favorecendo a gestão racional do uso de energia eléctrica e tendo ainda em mente a emergência *a posteriori* do standard PoEP (Power Over Ethernet Plus) [Di Minico 2005] para viabilizar a alimentação de um posto de trabalho com recurso apenas à tomada de rede, permitindo assim que as funções básicas de comunicação continuem a operar em caso de falha de corrente.
- **Convergência funcional de comunicações**. Ao consolidar os recursos típicos existentes no posto de trabalho, colapsando as funções de telefone (PBX ou VoIP), videofone e computador num só dispositivo (Figura 3.12).
- **Design inteligente**. O sistema deverá estar pensado de modo a ser o mais compacto e ergonómico possível, com um mínimo de peças móveis e níveis reduzidos de ruído.
- **Interoperabilidade**. O sistema deverá ser pensado numa perspectiva ecuménica em termos de serviços e protocolos suportados devendo, por exemplo, ser capaz de aceder a partilhas SMB/Windows ou utilizar impressoras locais ou remotas, independentemente do protocolo de rede que estas utilizem.

  Deve também ser capaz de suportar um conjunto de hardware (periféricos e interfaces de expansão) consistente, privilegiando a fiabilidade sobre a excessiva variedade de componentes suportados.
- **Manutenção mínima e actualização fácil**. Do ponto de vista da gestão esta plataforma deve apontar ao ponto óptimo de equilíbrio entre a mobilidade e a liberdade dos utilizadores. Os *updates* do sistema operativo e aplicações deverão ser levados a cabo automaticamente aquando do arranque do sistema, através de um mecanismo simples que compare a versão local com a existente nos servidores, propondo ao utilizador a actualização do *firmware* (que poderá ser forçada, segundo opção do administrador do sistema). Contudo, a arquitectura proposta deverá especificar claramente uma política de ciclo de actualizações que force, tanto quanto possível, à adopção de prácticas de gestão correctas – como é o caso de testar quaisquer actualizações antes de fazer o seu deployment.
- **Robustez e resiliência**. A concepção deve procurar um equilíbrio entre a idempotência e a mobilidade – sem acesso fiável a dados remotos, não se pode implementar uma plataforma sem estado local que seja viável. Existe por isso uma clara necessidade de lidar com esta situação de uma forma o mais transparente possível – o sistema necessita de poder aceder aos dados do utilizador e permitir a este levar a cabo a sua sessão de trabalho com o mínimo possível de percalços. Tal só é possível se o sistema for concebido tendo em mente o paradigma de *disconnected computing*, sendo capaz de operar com condições de conectividade intermitente e fazendo uso de mecanismos de redundância quando estes existirem.

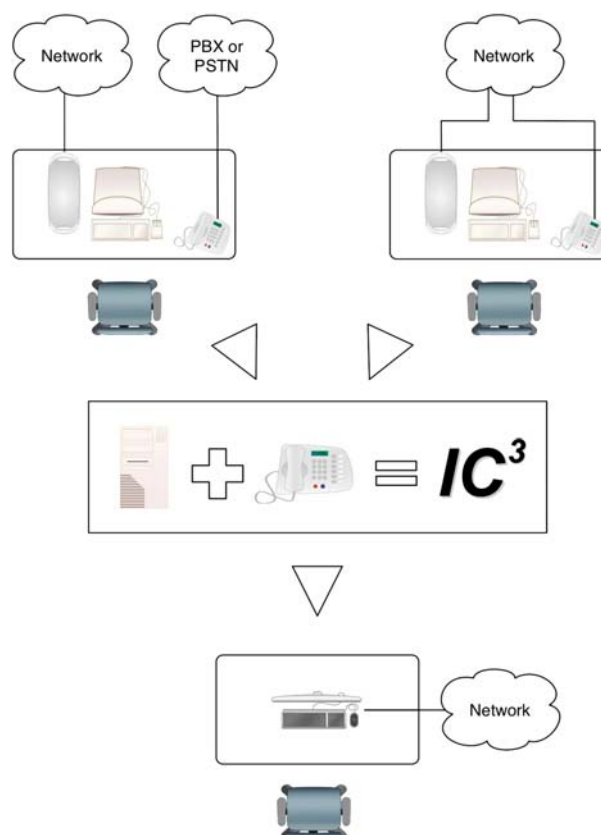



Figura 3.12 – Convergência de Funcionalidades na Platforma IC^3

Quando agrupadas, estas características constituem o sustentáculo prático do projecto IC^3, constituindo o ponto de partida para o trabalho e, de certo modo, a "ponte" que irá ligar o conceito da *Balanced Computing Platform* ao esboço da arquitectura que é apresentado no próximo Capítulo.

3.4.2 Projecto DOMUS: *Domestic Oriented Multiservice Self-Managed Appliance*

Para além do Projecto IC^3, foi desencadeado em paralelo um projecto para aplicar o conceito de *balanced computing platform* em ambientes domésticos e SOHO (*Small Office, Home Office*). A ideia é estender a arquitectura IC^3, com algumas contribuições vindas do Projecto OpenDMS (como o suporte PXE) de modo a criar uma gateway doméstica aberta, extensível, com baixos custos de manutenção e baseada em *hardware of-the-shelf* (Figura 3.13).

Esta *gateway* deverá ainda ser potencialmente capaz de integrar a parafrenália de serviços domésticos/SOHO potenciada pela Internet de banda larga, tais como acesso seguro à Internet, VoIP, Fax, Voice Mail, VcoIP, VoD, centros de entertenimento, televigilância, telemetria, domótica e serviços domésticos de impressão e de armazenamento de dados.

Estas funcionalidades são actualmente suportadas, no melhor dos casos, por pequenas caixas dedicadas tais como os *routers* de banda larga, com reduzida funcionalidade e flexibilidade. Novos serviços requerem frequentemente a instalação *on-site* de equipamentos adicionais, aumentando os custos de activação, e consequentemente, o TCO da infraestrutura. Além disso, os utilizadores finais necessitam de configurar, administrar e actualizar esses equipamentos, o que constitui um obstáculo à disseminação desses serviços.

A plataforma PC é suficientemente poderosa e flexível para integrar os serviços presentes e futuros num único equipamento. No entanto, os seus custos de manutenção são demasiado elevados, devido à complexidade da arquitectura PC, que padece sistematicamente dos mesmos problemas independentemente de falarmos do contexto de uma rede empresarial ou

de um ambiente doméstico. É por este motivo que o projecto DOMUS se propõe recorrer a soluções técnicas desenvolvidas no âmbito dos projectos IC³ e OpenDMS, pensados de raiz para dar resposta às questões envolvidas na gestão de PCs.

Resumidamente, o objectivo deste projecto é criar um sistema autónomo q.b., capaz de desempenhar as funções de *hub* doméstico de convergência digital de forma simples, autónoma e fiável.

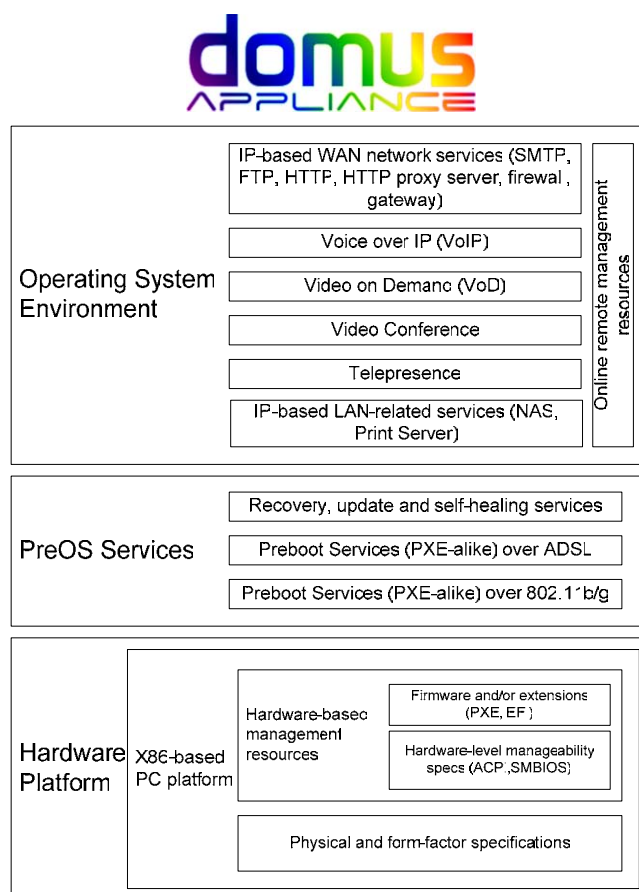



Figura 3.13 – Arquitectura DOMUS

3.5 Conclusão

Neste Capítulo foi apresentado o trabalho prévio do candidato na área de *desktop* management, substanciado no Projecto OpenDMS (Secção 3.1). Com base nas conclusões desse trabalho e na evolução entretanto registada na área em geral (Secção 3,2), foi identificado um espaço por preencher entre o modelo do PC clássico e do *Network Computer*. Foi assim proposto um novo modelo (Secção 3.3), designado por *Balanced Computing Platform*, que se espera venha a conciliar os aspectos mais interessantes de cada um desses dois paradigmas.

Na sequência dessa proposta, foram apresentadas as linhas gerais de dois projectos (Secção 3.4): o Projecto IC³ e o Projecto DOMUS. O primeiro, sobre o qual assenta o trabalho de investigação subjacente a esta dissertação, tem por objectivo apresentar uma interpretação da *Balanced Computing Platform* para ambientes corporativos. O segundo, que está agora a ser iniciado, propõe-se aplicar esses conceitos em ambientes domésticos e SOHO.

No próximo Capítulo será discutido em maior detalhe o Projecto IC³.

4. A Plataforma IC^3

A plataforma IC^3 tem como principal objectivo servir de prova de conceito à já mencionada *Balanced Computing Platform*, procurando articular os aspectos da gestão, fiabilidade, actualidade tecnológica e usabilidade – características que irão influenciar directamente o TCO da plataforma em função da medida do equilíbrio alcançado entre si.

Neste Capítulo será descrita a arquitectura da Plataforma IC^3, analisando-se diversos aspectos da sua concepção e desenvolvimento e procurando-se compreender como estes vão de encontro aos requerimentos funcionais enunciados no Capítulo 3.

A Secção 4.1 descreve, em traços gerais, um conjunto de requisitos suplementares e opções técnicas (em termos de *hardware* e *software*) adoptadas no sentido de concretizar e complementar o leque de funcionalidades previstas para a plataforma IC^3.

Na Secção 4.2 é discutida em detalhe a questão da selecção do *hardware* de suporte para o protótipo IC^3, procurando-se justificar as opções tomadas em função da satisfação dos requisitos funcionais previamente enunciados.

A Secção 4.3 aborda o tema dos requisitos técnicos considerados adequados para as necessidades do projecto em termos do ambiente de operação e *software* de sistema.

A Secção 4.4 trata de esclarecer o modo como se procedeu à integração entre o *software* de sistema e o *hardware*.

Partindo da estrutura delineada pelas secções anteriores, numa abordagem *bottom-up*, a Secção 4.5 completa o cenário com a descrição da operação do sistema em termos dos mecanismos de actualização, serviços e protocolos suportados, integração com mecanismos AAA (Autenticação, Autorização e *Accounting*) e aplicações.

As Secção 4.6 sintetiza o modelo proposto para a plataforma IC^3 numa perspectiva abrangente, mostrando até que ponto a arquitectura discutida ao longo deste capítulo vem ao encontro das necessidades e requisitos funcionais previamente enunciados.

Finalmente, a Secção 4.7 valida o conceito subjacente à plataforma IC^3 com base num ensaio levado a cabo num cenário real de utilização e integração da plataforma.

4.1 Plataforma IC³: Requisitos Suplementares

Os requisitos funcionais identificados para a plataforma exigem o recurso a um conjunto de tecnologias e conceitos que visam completar a função da *Balanced Computing Platform* nos moldes anteriormente descritos. O objectivo desta Secção consiste em mostrar de que forma as opções técnicas que foram tomadas procuram dar resposta aos requisitos anteriormente identificados (*cfr*. Secção 3.4.1):

- Idempotência da plataforma.
- Transparência no acesso a recursos e suporte para mobilidade.
- Eficiência *MIPS/Watt*.
- Convergência funcional de comunicaçõe.
- Design inteligente.
- Interoperabilidade.
- Manutenção mínima e actualização fácil.
- Robustez e resiliência.

A primeira consideração a ter em conta diz respeito aos objectivos especificos do protótipo IC³: não se pretende construir uma versão *high-end* da *Balanced Computing Platform*, mas sim um sistema com capacidades mais modestas e adequadas para um posto de trabalho típico. Seguindo esta orientação, seleccionou-se o seguinte conjunto de aplicações e componentes a incorporar na plataforma:

- Cliente de mail/*groupware/*PIM.
- *Browser* Web com *plugins* Java e Macromedia Flash.
- Suporte Java.
- Pacote *Office* completo.
- Cliente VoIP/SIP e videoconferência H.323 (no âmbito da satisfação do requisito da convergência de comunicações).
- Software de ilustração/edição de imagem de complexidade média.
- *Media Player* com suporte amplo de *codecs* (excepto HD) e DVD/MPEG-2.
- Software de *messaging*.
- Suporte de protocolos de *desktop* remoto RDP, VNC e X11.

O leque de aplicações a incorporar, aliado aos requisitos funcionais identificados, influencia de modo decisivo a escolha dos componentes de hardware do protótipo, uma vez que a capacidade de processamento requerida para operar estas aplicações de modo satisfatório é reduzida quando comparada com os níveis de desempenho oferecidos pelos *desktops* actuais. Esta constatação permite optar por arquitecturas mais simples, mais maduras, mais robustas e mais eficientes em termos energéticos, sem que a consequente redução de capacidade computacional comprometa o desempenho do sistema para os fins em vista.

Do ponto de vista da flexibilidade e maleabilidade da plataforma, procurou-se obviamente recorrer a opções tão normalizadas e abertas quanto possível, de modo a facilitar a posterior evolução e adaptação da plataforma. Deste modo, surgem dois requisitos adicionais, a juntar à lista previamente definida:

- **Recurso a *hardware* normalizado.** O protótipo deverá ser construído com *hardware* normalizado, evitando sempre que possível o recurso a componentes de formato ou especificação proprietária.

- **Recurso a *software,* protocolos e componentes aceites, abertos e disseminados.** A natureza deste sistema, eunquanto prova-de-conceito, não é compatível com a oferta existente de sistemas operativos proprietários da família *Windows*, nem com os esquemas de licenciamento envolvidos (que, por si só, podem potencialmente comprometer a viabilidade da plataforma). Assim, não resta outra alternativa senão enveredar pelo uso de *software* de sistema, serviços e aplicações abertas. Analogamente, considera-se como critério de escolha a conveniência em que todos os protocolos e normas a adoptar sigam a mesma filosofia.

Para satisfazer o objectivo da idempotência da plataforma, tanto o sistema operativo como o conjunto-base das aplicações deverão ser tratados como se fossem *firmware* actualizável. O meio físico de armazenamento deste *firmware* deverá estar inacessível ao utilizador final e os seus dados, perfis e correspondentes configurações deverão estarão armazenados em servidores de rede. Deste modo cada dispositivo constituirá, de facto, uma entidade idempotente, cuja substituição em caso de avaria poderá ser feita de forma imediata e sem perda de tempo.

Para corresponder aos restantes desafios, a plataforma deverá ser ainda dotada das seguintes características:

- **Autenticação uniforme,** operando de forma idêntica em LAN ou WLAN e devendo ser articulada com um serviço de directoria para permitir suportar de forma elegante o *roaming* de utilizadores.
- ***Roaming* de VoIP e suporte de Videoconferência.** Graças à articulação com o serviço de directoria, torna-se possível concretizar a existência de extensões telefónicas que acompanhem o percurso físico dos utilizadores, onde quer que estes se autentiquem, conseguindo que a infraestrutura de voz ganhe uma assinalável plasticidade. Analogamente, poderá ser obtido o mesmo efeito para o serviço de videoconferência recorrendo a um *gatekeeper* H.323.
- **Conectividade de rede robusta baseada em sistemas de ficheiros distribuídos.** A este nível importa referir as dificuldades específicas impostas pelas redes WLAN: não são sendo suficientemente fiáveis, dificultam a articulação entre dois requisitos essenciais: idempotência e mobilidade. Sem acesso fiável e robusto a dados remotos não é possível ter uma plataforma viável sem armazenamento de estado local. É por isso necessário lidar com esta situação de forma tão transparente quanto possível: mesmo em ambientes WLAN é necessário que o utilizador tenha acesso aos seus dados e possa conduzir a sua sessão de trabalho com o mínimo de percalços. Para isso, torna-se fundamental que o sistema seja concebido tendo em mente o paradigma de *disconnected computing*.

  A solução, a nosso ver, passa pelo recurso a um sistema de ficheiros distribuído – e com suporte a estrutura redundante – cuja concepção tenha em conta os problemas inerentes à computação móvel, suportando de forma implícita o modo de operação *disconnected*. Podendo operar em modo de desconexão intencional ou acidental, esta solução permitirá que utilizadores continuem a trabalhar nos seus ficheiros até que a conectividade seja restaurada (existindo ainda a possibilidade de salvaguarda de dados recorrendo a gravadores de CDs ou *drives* USB *flash* nos casos em que a sessão termine antes que seja restaurada a conectividade).

  O recurso a este tipo de sistema de ficheiros permite um considerável grau de resiliência, graças à potencial redundância da infraestrutura de servidores. É possível ter um ou mais servidores a fornecer a mesma informação, viabilizando a replicação da infraestrutura com boa escalabilidade. Refira-se ainda que sistemas de ficheiros distribuídos mais clássicos, como o NFS e o SMB, deverão também ser suportados em nome da interoperabilidade, não obstante carecerem de mecanismos de robustez adequados aos postos de trabalho móveis.

Tomando todos estes aspectos em consideração, nas restantes Secções deste capítulo discutirão a plataforma IC³ seguindo uma abordagem *bottom-up* – começando no nível do hardware e terminando nos serviços suportados – procurando esclarecer e justificar, passo-a--passo, as opções tomadas em cada aspecto específico e a forma como estas encaixam nos requisitos identificados.

4.2 Opções de *hardware*

Desde o início que se tornou claro que a solução de conjunto encontrada deveria seguir, em diversos aspectos, o sistema que foi inegavelmente a inspiração deste protótipo: o já mencionado IBM PS/2 Energy Desktop – uma máquina compacta, sólida, eficiente em termos energéticos, versátil e ergonomicamente correcta. Por outro lado, para satisfazer os objectivos traçados pela especificação funcional, o protótipo deve ser construído com hardware normalizado, evitando sempre que possível o recurso a componentes de formato ou especificação proprietária (ao contrário do IBM PS/2 "E") de modo a facilitar o design e manutenção da plataforma. Ao recorrer a equipamento COTS (*commercial off-the shelf*) evita-se à partida a dependência de um conjunto restrito de fornecedores OEM (*Original Equipment Manufacturer*), mantendo controlada a *Bill of Materials* (BOM). Nesta Secção abordar-se-ão os aspectos mais pertinentes das opções que caracterizam o design do protótipo IC³, no respeitante aos componentes de hardware.

4.2.1 Arquitectura

A primeira questão que se colocou foi qual a arquitectura a adoptar. Seguir as arquitecturas dominantes em termos de eficiência energética – nomeadamente a ARM [ARM] e a MIPS [MIPS] – implicaria recorrer a componentes exóticos, aumentando as dificuldades de integração do software e reduzindo fortemente a capacidade computacional do sistema. A fim de obter um equilíbrio correcto entre capacidade de processamento, consumo energético e disponibilidade de equipamento, a decisão pendeu claramente a favor da família *x86*. Entre as opções existentes dentro dessa família, ponderaram-se as seguintes arquitecturas:

- **AMD Geode LX/ /GX,** derivada da família *MediaGX*, originalmente desenvolvida pela *Cyrix Semiconductors* e que, após uma passagem pela *National Semiconductors* [National], viu o seu *brevet* ser vendido à *AMD* [AMD]. As principais diferenças em relação ao *design* do *Geode* original centram-se no subsistema de memória, na *cache* de nível e na velocidade. As duas declinações desta família mais adequadas para um possível protótipo seriam a *GX533@1.1W* e a *LX800@0.9W*, respectivamente. Contudo, é bastante díficil encontrar *motherboards* em formatos normalizados para estes processadores a preços aceitáveis. Adicionalmente, os indicadores de desempenho disponibilizados pelo fabricante apoiam-se em *benchmarks* duvidosas (desenvolvidas de propósito para o processador) que tomam por referência um *design* Centaur/VIA (*core Samuel II*) de 2000/2001, cuja performance era inferior à de um CPU *Intel Celeron 400*. Para agravar a situação, a arquitectura não sofreu evoluções em aspectos como o subsistema VGA desde a sua incepção, em 1997.

- **Intel Pentium-M.** Derivada da *core P6*, mais precisamente da declinação que deu origem ao *Pentium III*, esta família prima por conjugar níveis de eficiência energética respeitáveis com elevado desempenho. Contudo, eficência energética não é sinónimo de baixo consumo, e os cerca de 45W necessários para um sistema completo (*cfr. caixa na página seguinte*) colocam-na por ora fora de questão. A opção pela variante ULV (*Ultra Low Voltage*) desta família de processadores poderia ser tomada em consideração, não fosse a ausência de *motherboards* em formatos normalizados a preços aceitáveis. Especula-se que com a nova geração de processadores *desktop* desta família (com o *codename Yonah*), até agora posicionada para o *mobile computing*, o problema da escassez e preço dos componentes seja resolvido em breve.

Consumo energético aproximado medido e calculado, por arquitectura

		EPIA 5000 – C3 533MHz			MII 10000 – C5P 1GHz		
		Vr (V)	**I(A)**	**Energia (W)**	**Vr (V)**	**I(A)**	**Energia (W)**
Barramento de alimentação	3.3V	3,27	2,02	6,59	3,33	2,44	8,13
	5V	4,20	1,22	5,12	5,01	3,02	15,13
	5VSB	4,55	0,04	0,17	4,95	0,05	0,25
	12V	12,14	0,07	0,79	12,00	0,20	2,40
Energia Total (W)				12,68			25,90
Envelope total do sistema (W)				**17,30**			**31,41**

		MII 12000 – C5P 1.2GHz			EPIA-N - C5P 933MHz		
		Vr (V)	**I(A)**	**Energia (W)**	**Vr (V)**	**I(A)**	**Energia (W)**
Barramento de alimentação	3.3V	3,33	2,44	8,13	3,20	2,97	9,50
	5V	5,01	3,42	17,13	4,64	2,38	11,04
	5VSB	4,95	0,05	0,25	4,76	0,13	0,62
	12V	12,00	0,20	2,40	11,88	0,15	1,78
Energia Total (W)				27,91			22,95
Envelope total do sistema (W)				**33,42**			**28,05**

		Celeron-M 1.5GHz			Pentium IV 3.0GHz		
		Vr (V)	**I(A)**	**Energia (W)**	**Vr (V)**	**I(A)**	**Energia (W)**
Barramento de alimentação	3.3V	3,37	3,12	10,51	3,37	3,61	12,17
	5V	4,89	2,51	12,27	5,06	1,77	8,96
	5VSB	5,02	0,14	0,70	5,04	0,06	0,29
	12V	12,51	1,30	16,26	11,96	5,84	69,85
Energia Total (W)				39,75			91,26
Envelope total do sistema (W)				**45,13**			**96,83**

	Carga no Barramento			
I (A)	3,3V	5V	5VSB	12V
Leitor de CD Slim	0,00	1,00	0,00	0,00
Cartão Compact Flash	0,00	0,10	0,00	0,00
Total (A)	0,00	1,10	0,00	0,00

- **VIA Eden C3 *Ezra* e Eden C3 *Nehemiah*.** A primeira destas famílias de processadores *(Ezra core)* é derivada de um *design* de origem *Centaur* (subsidiária da *VIA Semiconductors*) datado do início da década 2000 que foi lançado com o nome de *VIA Cyrix III* (ainda que o *design* nada tenha a ver com o portfolio da *Cyrix*, entretanto também adquirida pela *VIA*). A segunda família (*Nehemiah core*) é mais sofisticada e deriva de uma *core* de origem *Centaur*, a *C5X*. A declinação *C5P* incorpora um motor de encriptação – designado por *VIA Padlock* – que implementa o

algoritmo AES no *stepping* 8 do processador e os algoritmos de *hashing* SHA1 e SHA256 nas encarnações mais recentes (core *Esther*).

Esta família inclui processadores disponíveis em frequências que vão de 533MHz até 1,3 GHz, com consumos entre 7W e 20 W e com variantes dissipadas activa e passivamente. O *total design envelope* para os propósitos considerados na plataforma IC^3 varia entre os 17W e os 34W, aproximadamente (cfr caixa). Com desempenho ao nível, no melhor dos casos, de um processador *Intel Celeron 800*, beneficia ainda da disponibilidade de *motherboards* em formatos normalizados e a preços acessíveis.

Acresce ainda como argumento extra a capacidade de encriptação AES por *hardware* nas *cores C5P* e superiores, permitir atingir velocidades da ordem dos 1,7GB/s em modo ECB (*Electronic Codebook*) com blocos de 8kB. Isto permite, por exemplo, aliviar quase na totalidade o *overhead* do uso das bibliotecas de SSL e operar practicamente em *wire-speed* em ligações de rede local a 100Mbit, oferecendo assim níveis de desempenho muito favoráveis a ambientes VPN.

Considerando o binómio energia/desempenho, determinou-se que o protótipo IC^3 deveria basear-se na arquitectura *VIA C3*, sem dúvida, a mais adequada aos objectivos traçados a curto prazo. Prevê-se que a próxima geração desta família, denominada C7 (*core Esther*), opere entre 1.5 GHz e 2.0 GHz, consumindo entre 12W e 20W (*total design power* máximo do processador) com uma diferença de desempenho que se estima ser apenas 10% inferior a um *Pentium-M* com idêntica frequência de operação.

4.2.2 *Form-factor* e Dimensões Físicas

A segunda opção a tomar, o *form-factor*, surgiu quase como consequência directa da escolha da arquitectura. Para satisfazer a necessidade de um sistema compacto mas com alguma capacidade de expansão, a opção pendia claramente a favor do Mini-ITX [VIA 2001]. O Mini-ITX é um formato físico desenvolvido pela *VIA* no início desta década que limita o tamanho das *motherboards* a uma área de 17x17cm. A existência da família C3 encontra-se intimamente ligada aos *form-factors* Mini e Nano-ITX (12x12cm), uma vez que a maior fatia do seu volume de comercialização é feita em conjunto com este tipo de *motherboards* – não faria sentido oferecer um processador *low-profile* para outras aplicações que não os sistemas embebidos/*low-end*. Contudo, foram as características do formato Mini-ITX – preço acessível, baixos requisitos energéticos, tamanho reduzido e versatilidade – que permitiram a muitos entusiastas desenvolver todo o tipo de aplicações *Do-It-Yourself*, desde *media centers* até sistemas de *car-audio*.

O *form-factor* Mini-ITX compara-se favoravelmente com outros existentes no respeitante às dimensões, permitindo a concepção de *chassis* extremamente compactos (Figura 4.1), tal como aquele que foi usado para o protótipo IC^3. Estes *chassis* permitem inclusivamente rivalizar em tamanho com o *Winterm* estudado no Capítulo anterior.

Seguindo estas orientações, adoptou-se para o protótipo IC^3 uma *motherboard* Mini-ITX de *design* integrado, equipada com um processador *VIA C3 Nehemiah* (*C5P*) e incorporando ainda o suporte para audio, rede e video. O *chassis* escolhido é de origem *Morex*, mais precisamente o modelo *Cubid 3688* (Fig 4.1) com dimensões externas de 65x258x210 (Altura x Profundidade x Largura, em mm).

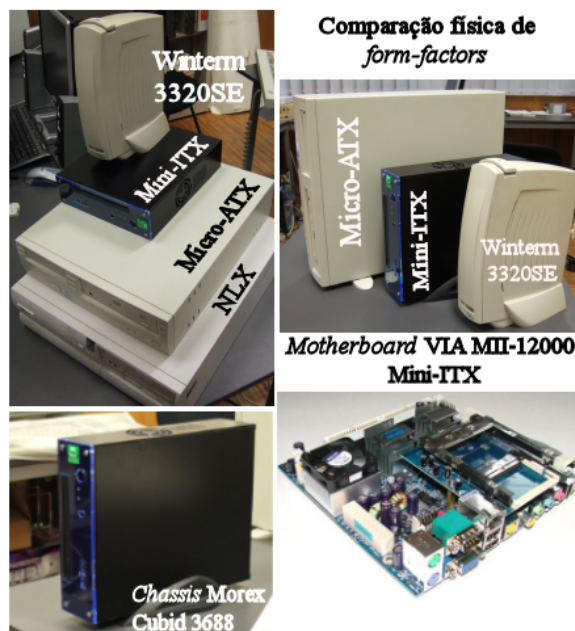



Figura 4.1 – Comparação Física de *Form-Factors*

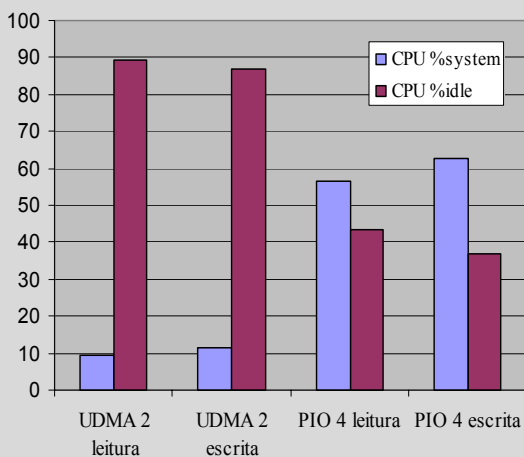
4.2.3 Armazenamento de massa local

Dados os requerimentos específicos de armazenamento do ambiente de operação do sistema, optou-se por um meio *solid-state,* nomeadamente por cartões *compact flash* (CF) – uma escolha racional do ponto de vista da eficiência energética, custo, disponibilidade e desempenho. Os cartões CF possuem actualmente capacidades que podem ir de 16MByte a mais de 20 GByte e incorporam uma interface PCMCIA IDE/ATA capaz de operar em modo PIO 4 (*Programmed I/O*) com débitos até 16,6MB/s (caso o processador consiga suster este débito, uma vez que é este que gere a transferência de dados) nas normas CF 2.0 e UDMA 2/4 (permitindo *bursts* de 33,3/66.6 MB/s) na recentemente aprovada versão 3.0. Ao escolher o cartão mais adequado consideraram-se três factores:

- **MTBF** a par com o resto dos componentes (idealmente acima, o que não será díficil face à natureza da memória *Flash*).
- **Opção pela norma CF 3.0.** Os cartões CF mais difundidos são conformes com a norma CF 2.0, sendo capazes de taxas reais de transferência na ordem dos 3-6MB/s em leitura e 2,5-5MB/s em escrita, com tempos de acesso entre os 0,7 e 1,5ms, operando em modo PIO, com o inconveniente de recorrer à CPU para as operações de I/O. Ainda que os cartões CF vulgares apresentem desempenho satisfatório, a opção por modelos conformes com a norma CF 3.0 acarreta claros benefícios, nomeadamente no respeitante aos tempos de acesso, ao protocolo de transferência de dados e aos débitos suportados (*cfr. caixa*).
- **Capacidade mínima de 512 Mbyte**, determinada em função da aplicação em causa.

É importante mencionar uma limitação inerente à memória *flash*: a existência de um máximo de operações de escrita por sector (chamado de *endurance limit*, rondando os 10.000.000 ciclos de escrita ou mais por sector, graças à tecnologia das *flash* de tipo NAND e aos algoritmos de *write relocation* e *wear level* que ajudam a distribuir o desgaste de forma uniforme ao longo do volume de memória). Contudo, essa limitação perde relevância no contexto do IC^3, pois o volume primário do sistema irá operar em modo *read-only* (excepto no caso de actualizações do sistema).

Os cartões *Compact Flash* 3.0

Os cartões produzidos conforme a norma CF 3.0 incorporam um interface PCMCIA (*16 bit*) IDE/ATA com suporte para os modos UDMA 2 (33MB/s) e 4 (66MB/s).
No caso do modelo de cartão específico que se utilizou no IC³ – o *Pretec Cheetah* 80x – anunciam-se taxas de transferência sustentadas em operações lineares da ordem dos 13MB/s em leitura e 12MB/s em escrita, com um MTBF de 1.000.000 horas. As medições efectuadas provam que este cartão é capaz de operar com taxas de transferência entre os 9,6 e os 11MB/s em leitura, e os 5,0 e os 9,5MB/s em escrita, com um tempo de acesso médio da ordem dos 0,9ms.
Graças ao suporte UDMA, a transferência dos dados de e para a memória processa-se através do controlador IDE/ATA, que toma conta do processo após ter sido instruído para tal sem necessidade de envolver o processador, sendo esta apenas notificado no fim da operação. Este modo de operação liberta o processador do *overhead* de I/O, permitindo reduzir em 50%-60% a carga do sistema durante as operações de acesso à memória.

Um estudo de desempenho* efectuado com o cartão seleccionado (ver Gráfico), operando em ambos os modos – PIO 4 e UDMA 2 – demonstra claramente a vantagem do último modo de operação em termos de consumo de CPU, permitindo tempos de carga de programa mais curtos e um funcionamento mais fluído em situações de I/O intenso.

**O teste foi levado a cabo num sistema EPIA MII-10000 com 512MB de memória e uma instalação de linux (kernel 2.4.19), utilizando para o efeito o utilitário "dd" para as operações de leitura e escrita (criação e leitura de volumes de 500MB em blocos de 512 bytes) com o comando "vmstat 2" a correr em background. Cada bateria de testes incluiu 40 ensaios.*
Modos de transferência de dados configurados da seguinte forma:
UDMA 2: hdparm –d1 –c0 –X66 /dev/hda
PIO modo 4: hdparm –d0 –X12 –c0 /dev/hda

Acrescente-se ainda que o cartão CF é instalado no protótipo IC³ com o auxílio de um adaptador CF-IDE (Figura 4.2) com suporte para DMA (conectando o pino 21 do *interface* IDE ao cartão) que permite a ligação directa ao interface IDE da *motherboard*, sendo detectado e utilizado como um disco rígido convencional.

Figura 4.2 – Adaptador CF-IDE

Alternativamente, poderia ser utilizado um adaptador CF-IDE duplo (Figura 4.3), permitindo a existência de um segundo cartão – um *shadow card* – que poderá ser utilizado para diversos fins. Por exemplo:

- Manutenção de uma cópia intacta do ambiente de sistema, para permitir operações de recuperação (*fallback*) em caso de falha física do cartão primário ou falha no processo de actualização, com impossibilidade de acesso aos servidores de actualização.
- Auxílio no processo de actualização (podendo descarregar-se a nova imagem de sistema para o cartão secundário, que seria activada apenas depois de confirmado o sucesso da operação e efectuada uma análise de integridade).

Figura 4.3 – Adaptador CF-IDE Duplo

4.2.4 *Design* Final do Protótipo IC^3

Integrando-se os componentes identificados nas Secções anteriores, foi possível chegar ao *design* final do protótipo IC^3 (Figura 4.4).

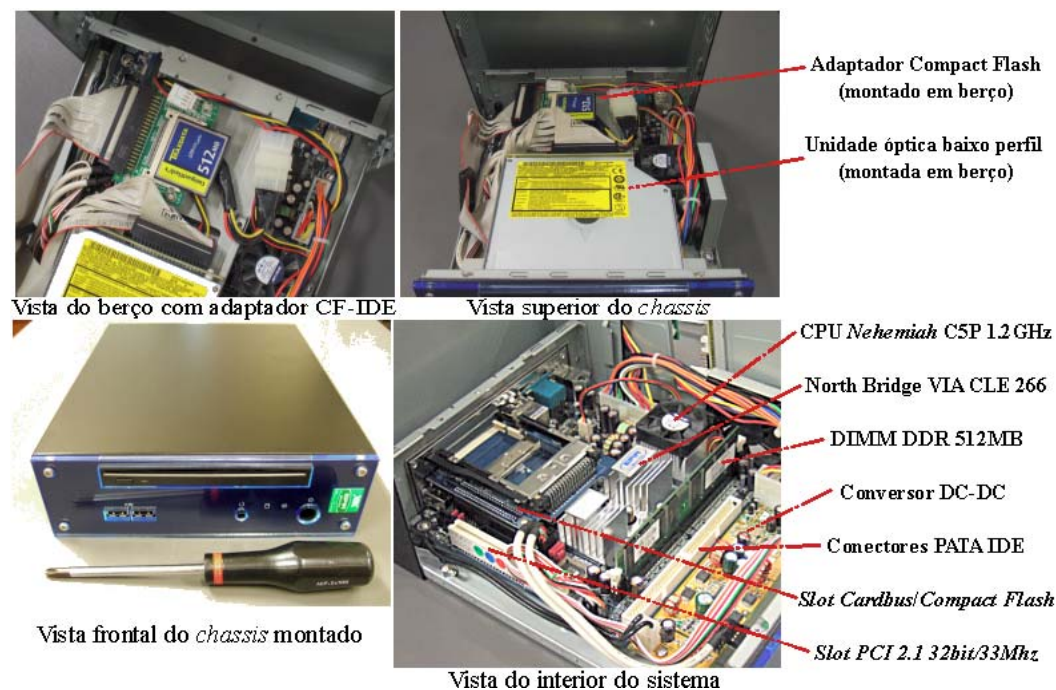



Figura 4.4 – Componentes de Hardware do Protótipo IC^3

O protótipo assenta numa *motherboard VIA EPIA MII-12000*, dotada de um processador *VIA C5P* a 1,2GHz e incluindo audio integrado de 6 canais (*codec VIA VT1616 AC'97* de 6 canais), interface de rede *Fast Ethernet* (*VIA VT6103*), 1 porta paralela de 25 pinos, 1 porta série de 9 pinos, uma porta *Firewire*/IEEE1394 400Mbps (*VIA VT6307S*), 4 portas USB 2.0 (480Mbps), 2 portas PS/2, 1 conector VGA analógico de 15 pinos, 1 saída RCA configurável para vídeo composto ou SPDIF e uma saída S-Vídeo.

A nível de expansão, encontra-se disponível um *slot* PCI conforme com a versão 2.1 da norma, um slot *CardBus* e um slot *CompactFlash*, controlados por um ASIC *Ricoh* (R5C476II/R5C485). O *chipset* da *motherboard* é de origem VIA (VIA *CastleRock* CLE266) sendo capaz de suportar um máximo de 1GByte de memória (1 DIMM DDR SDRAM, 512MByte instalados no protótipo) com um FSB de 266MHz, recorrendo a uma arquitectura UMA (*Unified Memory Architecture*), em que o subsistema de vídeo recorre a memória do sistema para operar. O subsistema de vídeo consiste no VIA/S3 *Unichrome* integrado na *North Bridge* e com aceleração MPEG-2 e 3D (se bem que esta última seja algo limitada). A *South Bridge* (VIA VT8235) suporta os 2 interfaces PATA existentes, compatíveis com os modos de transferência da gama PIO (1 a 4) e UDMA (1 a 5).

Para efeitos de armazenamento de massa o protótipo está equipado com um cartão *compact flash* 3.0 de 512MB de 80x *Pretec Cheetah*, ao qual ainda se acrescentou um *combo* gravador de CD/leitor de DVD de baixo perfil *slot-in* de origem Panasonic (CW-8124B) para fornecer um meio de acesso a dados amovível.

Os valores de MTBF anunciados por componente são os seguintes:

- **Motherboard:** 70.000 horas (fonte: *VIA Platform Systems Division*).
- **Fonte de alimentação:** 100.000 horas (fonte: *Morex*; valor projectado de 120000 horas em operação contínua a 25ºC, carga máxima).
- **Cartão CF:** 1.000.000 horas (projectado, *POH – Power On Hours*).
- **Unidade óptica:** 60.000 horas (fonte: *Panasonic*).

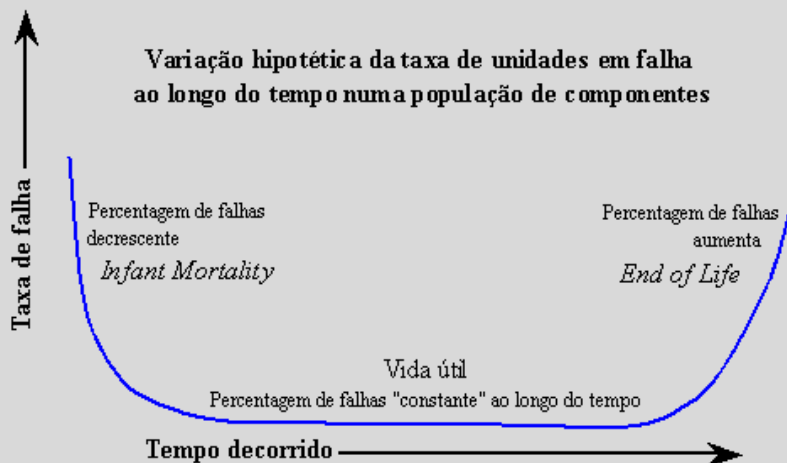
Os números aqui apresentados, excepto no referente aos cartões CF, são valores de MTBF operacionais e não projectados, dado que estes componentes têm já tempo de comercialização suficiente para recolher, analisar e apresentar valores reais com algum significado, baseados na observação do comportamento de amostras retiradas dos lotes já em circulação e utilização (*cfr. caixa na página seguinte*).

Deste modo, o conjunto de opções técnicas aqui descritas (algumas herdadas dos sistemas embebidos) permitiram construir uma plataforma pensada de raíz para a "optimização dos três M's": MTBF, MTTR e MTBCF – um sistema robusto, fiável, expansível, energeticamente eficiente e ergonómico. Procurou-se assim satisfazer diversos requisitos funcionais enunciados na Secção 3.4.1: fiabilidade, robustez e ergonomia/design inteligente. O recurso a poucas partes móveis permite também reduzir os pontos de falha e o nível de ruído da plataforma, estimado em 25dBA a um metro (Tabela 4.1). Note-se por último que seria possível ajustar o protótipo – mantendo os as mesmas escolhas fundamentais – de modo a obter níveis de ruído ainda mais reduzidos.

O Significado do MTBF

O MTBF é uma métrica cujo real significado permanece desconhecido para muitos, sendo muitas vezes utilizado para efeitos de *marketing* enganoso. Entenda-se, a bem da clareza, que um MTBF de 50.000 horas não equivale a 5,7 anos (50.000/((365*24)+6)) de funcionamento ininterrupto. Para melhor compreender o conceito é preciso saber de que forma é determinado o MTBF.

É geralmente aceite o facto da taxa de falhas durante a vida de uma população de componentes não ser constante, sendo reconhecidas três fases: a fase inicial, onde o valor é alto (*infant mortality*, por exemplo devido a componentes *dead on arrival* e/ou falhas no controle de qualidade); a fase de amadurecimento, onde os níveis são baixos e constantes (*constant failure rate*); e a fase final, onde o valor aumenta novamente (*wearout* perto do final de vida). Este ciclo corresponde a uma curva conhecida como a *Bathtub Curve*.

O MTBF corresponde ao inverso da taxa de falha na fase intermédia de vida do componente (quando temos um *constant failure rate* – ao qual se aplica uma distribuição exponencial) sendo este medido em unidades de componentes-tempo por falha, i.e. falhas por máquinas-hora ou sistemas-ano.

Existem valores de MTBF que excedem brutalmente o *wearout time*. Uma população com uma taxa de falha de 2 componentes por cada 1000 unidades-ano tem um MTBF de 500 anos (na realidade, 500 componentes-ano por falha), mas a imensa maioria dos componentes estará em falha muito antes desse tempo ser atingido.

O MTBF é uma característica extraída de populações e que apenas se pode aplicar a populações. Assim sendo, é indicado para estimar, por exemplo, quantos *spares* de determinado componente são necessários para um parque de PCs. É no entanto inapropriado para determinar a esperança do ciclo de vida (*service life*) de um indivíduo isolado da população.

Referências	dBA
Limite da audição humana	0 dBA
EPIA CL6000 (Mini-ITX *Fanless*)	0 dBA
Respiração	10 dBA
Murmúrio a 1 metro	20 dBA
Protótipo IC³ (VIA MII12000)	**25 dBA**
PC convencional	35 a 50 dBA
Chuva	50 dBA
Conversa	60 dBA

Tabela 4.1 – Nível de Ruído do Protótipo IC³

A escala dBA é logarítmica (base 10) e indexada à resposta sensorial do ouvido humano. Um incremento de 10 dBA representa um valor em dobro da resposta sensorial percepcionada pelo ouvido (loudness) e não do volume/intensidade (cujo incremento é de 10x)*, como é muitas vezes erradamente referido. Os valores são medidos à distância de 1 metro. Fonte: EPIA Operating Guidelines [VIA 2004b]*

4.3 Software de Sistema e Ambiente de Operação

Sendo tão importante como o *hardware*, o *software* de sistema é uma peça vital para a concretização do conceito. Aos requisitos funcionais identificados para a plataforma adicionou-se um conjunto de características adicionais que deverão ser tidas em conta na selecção de componentes de *software* a adoptar:

- **Capacidade de operação em modo *read-only***, maximizando a vida útil do meio de armazenamento de massa do volume do sistema (memória *Flash*) e reforçando a natureza *stateless* do conceito (cfr. Secção 3.4.1).

- **Suporte para Volumes Comprimidos**, com operações de leitura/descompressão em tempo real e *overhead* de CPU mínimo, permitindo assim incorporar o ambiente de operação dentro do limite dos 512MB do cartão *Compact Flash* seleccionado e minimizando a quantidade de dados a enviar via rede em caso de actualização do sistema.

- **Suporte de *Ramdrives***, de preferência dinâmicas, com *resize* em função da memória requisitada pelos programas sem nunca descer abaixo do limite do espaço já ocupado. Este tipo de *ramdrives* permite a operação do protótipo em modo de desconexão da rede, albergando as informações de estado de sessão absolutamente necessárias e críticas para o funcionamento do sistema em caso de ausência de conectividade. Este requisito deriva indirectamente dos requisitos funcionais da idempotência da plataforma e resiliência.

- **Suporte para *Stateless PnP***. De modo a permitir a idempotência da plataforma, não é armazenada localmente nenhuma informação do *hardware* de sistema e sua configuração, devendo este ser capaz de detectar, enumerar e configurar correctamente os componentes aquando do arranque, sem necessidade de informação de estado local.

- **Pilha Protocolar com Suporte QoS/CoS Adequado**. Esta necessidade está intimamente ligada com a convergência de comunicações, pois a solução apresentada pela plataforma IC^3 só será viável se se estiver assegurada a disponibilidade e qualidade das ligações de rede.

  Para melhor compreender este requisito, deve estabelecer-se um paralelo entre o sistema de telefonia tradicional e o baseado em IP – enquanto que, no primeiro caso, os *designers* do sistema consideraram preferível não haver comunicação a que esta seja estabelecida com deterioração de qualidade, por contraste, numa rede com comutação de pacotes (tipo IP) a disponibilidade de largura de banda pode ser incrementalmente comprometida enquanto não se atinja um ponto de ruptura total, continuando a permitir-se a admissão de tráfego, com consequente diminuição de débito e degradação da qualidade da ligação. Para que o objectivo da convergência de comunicações possa ser atingido, deverá ser possivel lidar de forma efectiva com os três maiores inimigos das comunicações em tempo real: a latência, a perda de pacotes e o *jitter* utilizando os mecanismos de CoS/QoS adequados e pertinentes do lado do cliente.

- **Suporte para Protocolo 802.1x**. De modo a permitir autenticação uniforme, determinou-se que o sistema operativo deveria suportar autenticação 802.1x com a *key frame* EAP-TTLS + EAPOL (*EAP over* LAN) e autenticação PEAP-MSCHAPv2. Juntos, estes procolos permitem o estabelecimento prévio de um túnel TLS (*Transport Layer Security*) para autenticação segura de forma independente do meio (LAN ou WiLAN). Esta opção permite também oferecer, à partida, segurança nas comunicações VoIP/Videoconferência, em meio *wireless*.

4.3.1 Sistema Operativo

A opção escolhida para o sistema operativo do protótipo recaiu sobre uma distribuição de Linux especificamente adaptada para este propósito. As razões pelas quais se tomou esta opção prendem-se com os seguintes aspectos:

- **Licenciamento (custos e propriedade intelectual).** A opção por um sistema operativo comercial para a construção do protótipo implicaria à partida o pagamento de uma licença. Mesmo recorrendo a licenças académicas, restariam ainda os problemas decorrentes da necessidade de adaptação do sistema para o fim específico a que se destina. Inevitalvelmente surgiriam questões de propriedade intelectual que poderiam condicionar de forma crítica a vocação aberta do conceito.
- **Amplo suporte de hardware, normas e protocolos.**
- **Adequação à tarefa.** O sistema operativo Linux dispõe dos meios, componentes e características necessárias para a conformidade com os requisitos funcionais enunciados nas Secções 3.4.1, 4.1 e 4.3.

  Dadas as capacidades modestas do *hardware*, o sistema operativo deverá ainda permitir tirar o melhor partido destas, maximizando o seu aproveitamento e rentabilizando o seu potencial. Existem mecanismos e recursos exclusivos do sistema operativo Linux (como o *compressed loop device*, descrito adiante) que lhe conferem uma excepcional plasticidade e cujo aproveitamento beneficia consideravelmente o protótipo, permitindo dar resposta aos requisitos funcionais e aos desafios directa ou indirectamente envolvidos nos processos de concepção e implementação.

  Como consequência directa desta característica, acrescem ainda benefícios ao nível de desenvolvimento, actualização e manutenção da plataforma – estando o ambiente de operação baseado em componentes cuja adaptação e/ou desenvolvimento é simples e possuindo os mecanismos (ou os meios para os implementar) necessários para possibilitar a concretização de um ciclo de teste e manutenção de actualizações prático e acessível.
- **Capital de conhecimento previamente adquirido.** Existe um capital de conhecimento adquirido ao longo do tempo que veio facilitar de forma decisiva o desenvolvimento da solução, devido a um elevado grau de familiaridade com o sistema Linux e suas especificidades.

Apesar da clara opção pelo *Linux*, deve esclarecer-se a bem da verdade, que não seria de todo impossível concretizar o protótipo com base em sistemas operativos de terceiros. A Microsoft, por exemplo, tem desde há algum tempo a família *Windows Embedded* [Microsoft 2006], constituída pelas variantes XPe ("e" de *embedded*), *POS* e *CE 5.0*, concebida especificamente para *appliances* e sistemas embebidos dos mais variados tipos (desde electrodomésticos até equipamentos para automóveis), sendo capaz de operar sobre volumes *read-only* (modo EWF – *Enhanced Write Filter* [Microsoft 2004b]). Estas variantes suportam um vasto conjunto de plataformas de *hardware* (o XPe apenas possui HAL para x86) com uma *footprint* que vai dos 4MByte até 40+MByte. De acordo com alguns rumores, já confirmados pela Microsoft, encontra-se mesmo em desenvolvimento um versão *light* do *Windows* (nome de código *Eiger*) [Foley 2005] que consistirá num híbrido entre as versões XP e XPe. Desprovida de diversos componentes não-críticos e direccionada para os mercados emergentes, esta versão será capaz de operar de forma satisfatória com *hardware* mais modesto e, pela sua vocação, poderia posicionar-se eventualmente como um potencial candidato para o projecto IC³.

Face às opções comerciais a principal vantagem do Linux prende-se com a facilidade com que este pode ser moldado às necessidades específicas da plataforma IC³, recorrendo a tecnologias e soluções sobejamente documentadas, abertas e acessíveis. Esta circunstância, combinada com o capital de conhecimento já existente, acarreta uma curva de aprendizagem e desenvolvimento bem mais estreita e acessível do que seria possível de outra forma.

4.4 Integração entre Ambiente de Operação e *Hardware*

Nesta secção discutem-se algumas questões relacionadas com a integração entre o hardware e o ambiente de operação do posto de trabalho. Será dada especial atenção a quatro questões: arranque do sistema, organização e operação do volume principal de armazenamento, manutenção da informação transiente da sessão, e mecanismos de *stateless plug-and-play*.

4.4.1: O arranque e inicialização do sistema

O arranque e inicialização do sistema, ilustrado na Figura 4.5, processa-se em três fases.

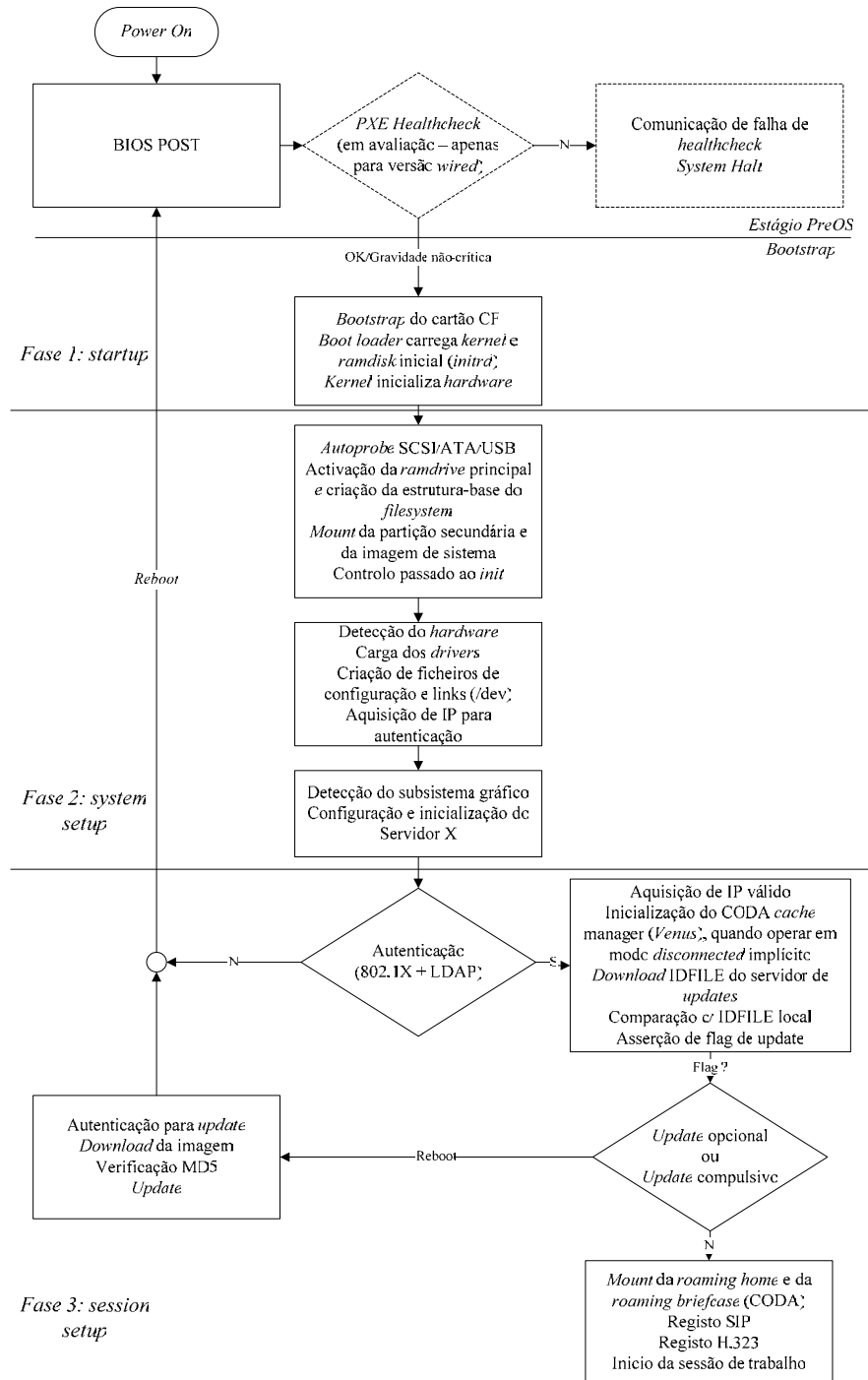
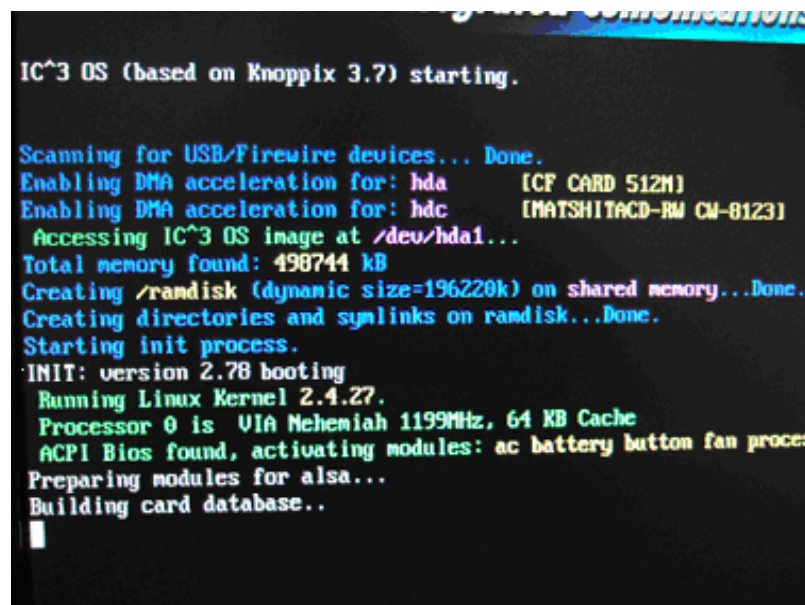



Figura 4.5 – Arranque e Inicialização do Sistema

Na **Fase 1 (*system startup*)**, posterior ao *powerup* do sistema, existem dois instantes críticos:

- No instante PreOS existe a possibilidade, em estudo, de recorrer a um agente derivado do desenvolvido para o projecto OpenDMS para verificação da sanidade do sistema antes do *bootstrap*. Devido a limitações de *firmware* não é possível dispor de suporte PXE para configurações *wireless*, pelo que esta característica apenas estaria disponível para sistemas *wired*.
- Na fase de *bootstrap* a BIOS passa controlo ao *Master Boot Record* (MBR) iniciando o processo de carga e execução do *boot loader* (o LILO – LInux LOader), que por sua vez se encarrega de carregar e descomprimir o *kernel* do sistema a partir da partição de arranque do cartão *Compact Flash*, dando início ao processo de inicialização do sistema operativo. É também da responsabilidade do LILO carregar o conteúdo da imagem *initrd* (*initial ramdrive* - *filesystem* montado numa *ramdrive* de dimensão fixa que contém or *drivers* e meios necessários para a inicialização do sistema, controlada através de um *script* chamado *linuxrc* que é executado por uma *shell* minimalista chamada *busybox*) para a memória do sistema.

Na **Fase 2 (*system setup*)** o conteúdo do ficheiro *initrd* é descomprimido e montado numa *ramdrive* contendo um *script* de inicialização, uma *shell* integrada (*busybox*) e um conjunto de programas e *drivers* vitais. Por intermédio do *script* de inicialização (*/linuxrc*) é levada a cabo a detecção dos dispositivos de armazenamento de massa existentes e a carga dos *drivers* necessários tendo em vista o acesso ao volume onde se encontra o ambiente de operação.

Uma vez criada a raíz do *filesystem*, inicializada a *ramdrive* dinâmica de sistema (com o valor máximo de 40% da memória principal, por razões que se esclarecem na Secção 4.4.3) e montada a imagem contendo o ambiente de operação, o *script* de inicialização levará a cabo a criação e disponibilização dos restantes elementos da estrutura de directórios e ficheiros do *filesystem* principal.

De seguida são criados os *links* e directorias que associam a estrutura do *filesystem*-base à *ramdrive* dinâmica e ao conteúdo da recém-montada imagem de sistema, passando em seguida o controlo ao processo *init* (como é tradicional nos sistemas tipo Unix). Neste estágio do processo de arranque é possível visualizar o ecrã apresentado na Figura 4.6.

```
IC^3 OS (based on Knoppix 3.7) starting.

Scanning for USB/Firewire devices... Done.
Enabling DMA acceleration for: hda      [CF CARD 512M]
Enabling DMA acceleration for: hdc      [MATSHITACD-RW CW-8123]
 Accessing IC^3 OS image at /dev/hda1...
Total memory found: 498744 kB
Creating /ramdisk (dynamic size=196220k) on shared memory...Done.
Creating directories and symlinks on ramdisk...Done.
Starting init process.
INIT: version 2.78 booting
 Running Linux Kernel 2.4.27.
 Processor 0 is  VIA Nehemiah 1199MHz, 64 KB Cache
 ACPI Bios found, activating modules: ac battery button fan proce
Preparing modules for alsa...
Building card database..
```

Figura 4.6 – Mensagens de arranque do sistema

No passo seguinte (Figura 4.7) é levada a cabo a detecção e configuração do restante *hardware* do sistema sob o controlo de um conjunto de *scripts* que constitui o sistema *PnP*

stateless (*scripts* esses adaptados dos existentes na distribuição adoptada), seguida da inicialização dos *interfaces* de rede. Seguidamente, e para finalizar este estágio do processo de arranque, é feita a configuração do subsistema gráfico (*X/Server*) e sua inicialização.

Figura 4.7 – Detecção do Hardware do Sistema

Na **Fase 3 (*session setup*)** é levada a cabo a autenticação do utilizador do sistema, com posterior configuração da pilha protocolar TCP/IP e verificação da existência de actualizações da imagem e correspondente nível crítico. De seguida pode ocorrer um de três cenários:

- O sistema efectua o registo SIP e H.323 e disponiliza o acesso à área de trabalho (*home*) do utilizador a partir de um servidor central, via rede, ou a partir de um meio local, caso se esteja a recorrer a uma *cache* persistente para suporte para mobilidade (cfr. Capítulo 5).

  Se o sistema utilizar o suporte implícito ao modo *disconnected* poderá ainda aceder a um volume contendo a *roaming briefcase* do utilizador, disponível num ou mais servidores, através do protocolo Coda (assunto discutido no Capítulo 5).

- Ao tomar conhecimento da existência de uma versão mais recente da imagem de sistema, o utilizador decide iniciar o processo de actualização.

- Existe uma actualização de nível crítico da imagem do sistema, pelo que este procede imediatamente ao *update*, de forma compulsiva

4.4.2 O volume principal do sistema: armazenamento, organização e operação

De acordo com o enunciado na Secção 4.2.3, o ambiente de operação do sistema encontra-se armazenado num cartão *Compact Flash* de 512MB com duas partições definidas (Figura 4.8):

- **Partição de arranque primária (*boot block*)**, onde se encontram o núcleo do sistema operativo, os binários essenciais, o segundo estágio do *boot loader*, uma imagem do tipo *initrd* (*initial* ramdrive) contendo os *drivers* essenciais e uma *shell* minimalista (*busybox*). Esta partição contém, adicionalmente, os meios necessários para efectuar o *update* do ambiente de operação ou a recuperação deste em caso de ocorrência de falha durante a actualização da imagem principal da partição de sistema (mini-ambiente embebido baseado em *Linux*).

- **Partição secundária de sistema**, onde se encontram dois ficheiros, um deles com o ambiente de operação (*filesystem* embebido do tipo *cloop*) e o outro (IDFILE) com

informações relacionadas com a versão da imagem de sistema e sua integridade (*hash* MD5).

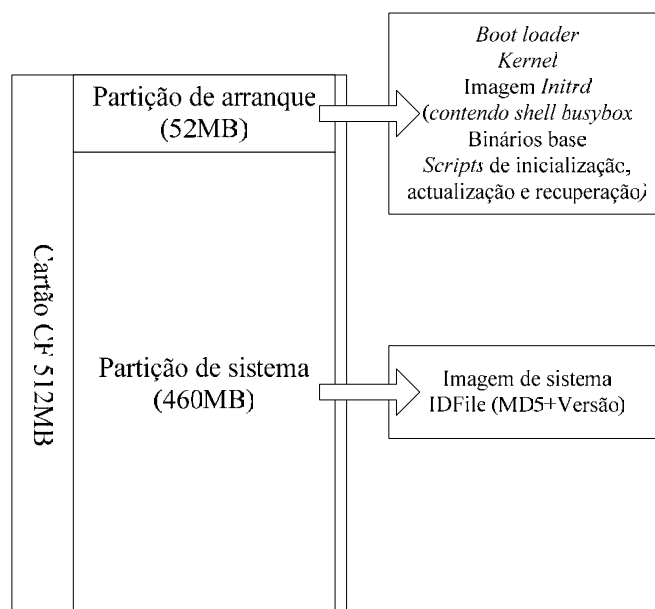
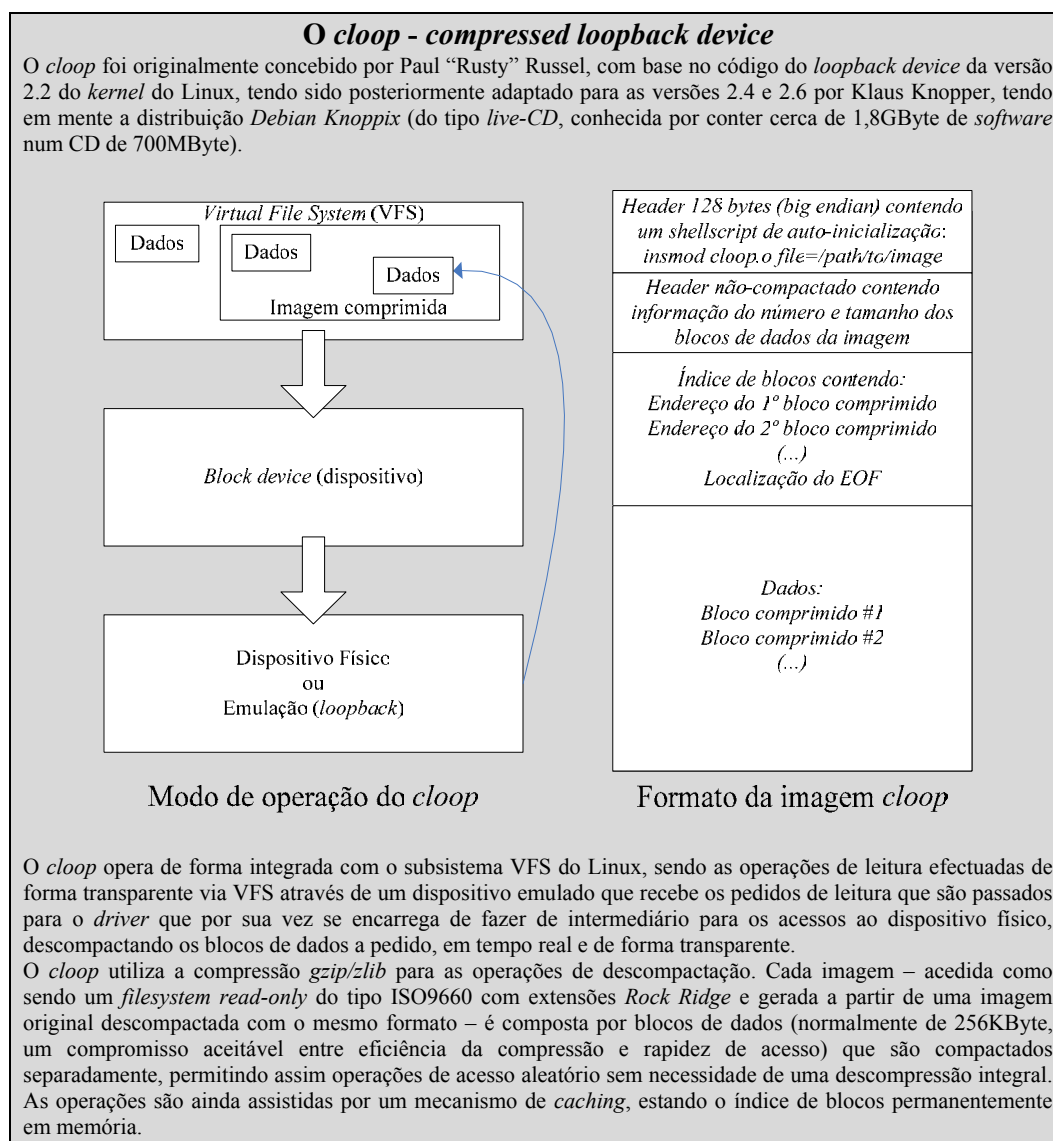



Figura 4.8 – Particionamento do Cartão Compact Flash

Cada partição contém um *filesystem* do tipo *ext2fs* (*second extended file system*), parametrizado com uma reserva de blocos para *superuser* de 1% (ao invés dos tradicionais 5%) e com a opção "*–Tlargefile4*" (um *inode* por cada 4MByte), para maximizar o espaço utilizável. A opção por este tipo de *filesystem* prende-se com a natureza do próprio volume de dados, que estará sujeito a poucas operações de escrita (apenas durante as actualizações), não carecendo portanto de mecanismos transaccionais como os dos *filesystems* do tipo *journaled.*

A imagem de sistema contém um sistema de ficheiros do tipo *cloop* (*compressed loop device*), contendo toda a informação num formato compactado, com taxas de compressão médias de cerca de 3:1, permitindo colocar cerca de 1,2GB de informação numa imagem de 410MB. Por esta razão, o *filesystem cloop* é uma característica de importância vital para a implementação do protótipo IC^3, na medida em que permite o armazenamento da totalidade do ambiente de sistema num único ficheiro, contendo uma imagem compactada com uma taxa de compressão razoável, pouco *overhead* de CPU nas operações de leitura e um elevado nível de tolerância a falhas (*cfr. caixa na página seguinte*).

Foi precisamente para minimizar o *overhead* de CPU nas operações de leitura, devido ao recurso a um volume compactado, que se optou por usar um cartão CF 3.0 com suporte DMA (visando a redução da penalização de I/O derivada do modo PIO e consequente agravamento imposto à operação do filesystem *cloop*), ainda que as especificações dos cartões CF 2.0 fossem aparentemente suficientes.

4.4.3 Operação em Modo *Read-Only* e Informação Transiente de Sessão

Uma das razões pelas quais não é comum encontrar um ambiente de operação capaz de funcionar sob um volume *read-only* (*cloop*) prende-se precisamente com a necessidade, por concepção, que a grande maioria dos sistemas operativos têm em recorrer a informação transiente e/ou persistente para o seu próprio funcionamento. A resposta para este problema passa por uma de três soluções:

- **Concepção, de raíz, de um sistema operativo** capaz de operar com em modo *read-only* no volume principal de sistema. Esta opção foi posta de lado, dado o tempo de desenvolvimento e recursos necessários para a sua concretização.

- **Adopção de um sistema já existente.** Não obstante a existência de sistemas comerciais concebidos para o mercado dos sistemas embebidos e que facilmente poderiam ser adoptados para este efeito, esta não é uma opção a considerar, visto colidir com os requisitos suplementares determinados na Secção 4.3.1.
- **Adaptação de um sistema já existente.** Esta solução é viável desde que se possa dispor de um determinado nível de controlo e capacidade de modificação do sistema operativo, o que pressupõe o acesso a aspectos de médio nível do sistema a modificar/adaptar. A adopção do *Linux* para sistema operativo, deve-se, em larga medida, ao facto de este possuir as características que permitem viabilizar esta opção.

A opção para o protótipo IC^3 recaiu sobre a última hipótese, recorrendo-se para o efeito à distribuição de *Linux Knoppix* (que opera em modo *read-only*, a partir de um CD-ROM, estando portanto a meio caminho do objectivo desejado). Para suportar a informação transiente de sessão recorre-se a uma *ramdrive* suportada por um tipo de *filesystem* do *Linux*, o *tmpfs*, que permite a criação de instâncias de tamanho variável (cfr. Secção 4.3).

O *cloop - compressed loopback device*

O *cloop* foi originalmente concebido por Paul "Rusty" Russel, com base no código do *loopback device* da versão 2.2 do *kernel* do Linux, tendo sido posteriormente adaptado para as versões 2.4 e 2.6 por Klaus Knopper, tendo em mente a distribuição *Debian Knoppix* (do tipo *live-CD*, conhecida por conter cerca de 1,8GByte de *software* num CD de 700MByte).

Modo de operação do *cloop*

Header 128 bytes (big endian) contendo um shellscript de auto-inicialização:
insmod cloop.o file=/path/to/image

Header não-compactado contendo informação do número e tamanho dos blocos de dados da imagem

Índice de blocos contendo:
Endereço do 1º bloco comprimido
Endereço do 2º bloco comprimido
(...)
Localização do EOF

Dados:
Bloco comprimido #1
Bloco comprimido #2
(...)

Formato da imagem *cloop*

O *cloop* opera de forma integrada com o subsistema VFS do Linux, sendo as operações de leitura efectuadas de forma transparente via VFS através de um dispositivo emulado que recebe os pedidos de leitura que são passados para o *driver* que por sua vez se encarrega de fazer de intermediário para os acessos ao dispositivo físico, descompactando os blocos de dados a pedido, em tempo real e de forma transparente.
O *cloop* utiliza a compressão *gzip/zlib* para as operações de descompactação. Cada imagem – acedida como sendo um *filesystem read-only* do tipo ISO9660 com extensões *Rock Ridge* e gerada a partir de uma imagem original descompactada com o mesmo formato – é composta por blocos de dados (normalmente de 256KByte, um compromisso aceitável entre eficiência da compressão e rapidez de acesso) que são compactados separadamente, permitindo assim operações de acesso aleatório sem necessidade de uma descompressão integral. As operações são ainda assistidas por um mecanismo de *caching*, estando o índice de blocos permanentemente em memória.

O *tmpfs* permite criar um *filesystem* não-persistente que utiliza a memória disponível até um patamar máximo, libertando-a quando esta não estiver a ser utilizada (*cfr. caixa*).

A *ramdrive tmpfs* é criada aquando do arranque do sistema, na segunda fase do processo, logo após o acesso à partição secundária do cartão CF – nesse mesmo instante é fixado um patamar máximo que garante que este nunca crescerá acima dos 40% da memória total (no caso do protótipo usado isto equivale aproximadamente a 191,6 MByte, considerando uma capacidade total de 480MByte correspondente a 512MByte de memória instalada, aos quais se subtraem 16 MByte para o subsistema gráfico UMA, 12,6 MByte para o *kernel* e respectivos *drivers* já carregados e 3,4 MByte para a ramdrive *initrd*) para evitar situações extremas onde este colida com as necessidades vitais do sistema.

Logo que o *tmpfs* esteja acessível, nele são criadas um conjunto de directorias com permissões de escrita (caso da */home* e */var*), ficheiros e *links* que associam as instâncias *unwritable* aos originais existentes na imagem do ambiente de sistema. Assim, consegue-se "iludir" o ambiente de sistema sem necessidade de modificações profundas na sua forma de operar, permitindo que este continue a criar/utilizar informação transiente relacionada com a sessão de trabalho sem necessidade de um suporte de armazenamento de massa com acesso *read/write*.

O *tmpfs* – *Virtual Memory filesystem*

O *Linux* dispõe, desde as primeiras versões do *kernel 2.4*, de um *filesystem* totalmente suportado sobre o sistema de gestão de memória virtual. O *tmpfs* permite a criação de um *filesystem* não-persistente, sem necessidade de formatação de um dispositivo (real ou não), cujos dados podem estar em memória ou algures em disco (em *swapped pages*) e cujo tamanho pode variar de forma dinâmica.

O *tmpfs* recorre ao mecanismo de paginação do sistema VMM (*Virtual Memory Manager*), requisitando páginas de memória para armazenar os ficheiros. Estas páginas estão sujeitas ao VMM que as move de e para o disco (*swapping*) de forma transparente para o *filesystem*. Assim, ao integrar-e com o VMM, torna-se possível variar de tamanho de forma dinâmica, libertando páginas quando estas não são necessárias (i.e., quando se apagam ficheiros) e requisitando-as quando necessário (até um limite máximo especificado no *mount* e que tem por objectivo evitar situações de *starvation*).
Num sistema normal, o *tmpfs* vai residir em memória durante a grande maioria do tempo, o que lhe confere um excelente desempenho, mesmo tendo em conta eventuais ocorrências de *swapping*. No caso do protótipo IC³ não existem meios de armazenamento de massa, pelo que existe a garantia de ter o *tmpfs* exclusivamente localizado em memória.

Assim, a estrutura de directorias do sistema ficará organizada como ilustra a Figura 4.9, sendo esta recriada a cada ciclo de *power-up* de forma articulada entre o volume de sistema e a *ramdrive tmpfs,* por forma a permitir a operação do ambiente de sistema a partir de um volume *read-only*, sem armazenamento de estado local persistente.

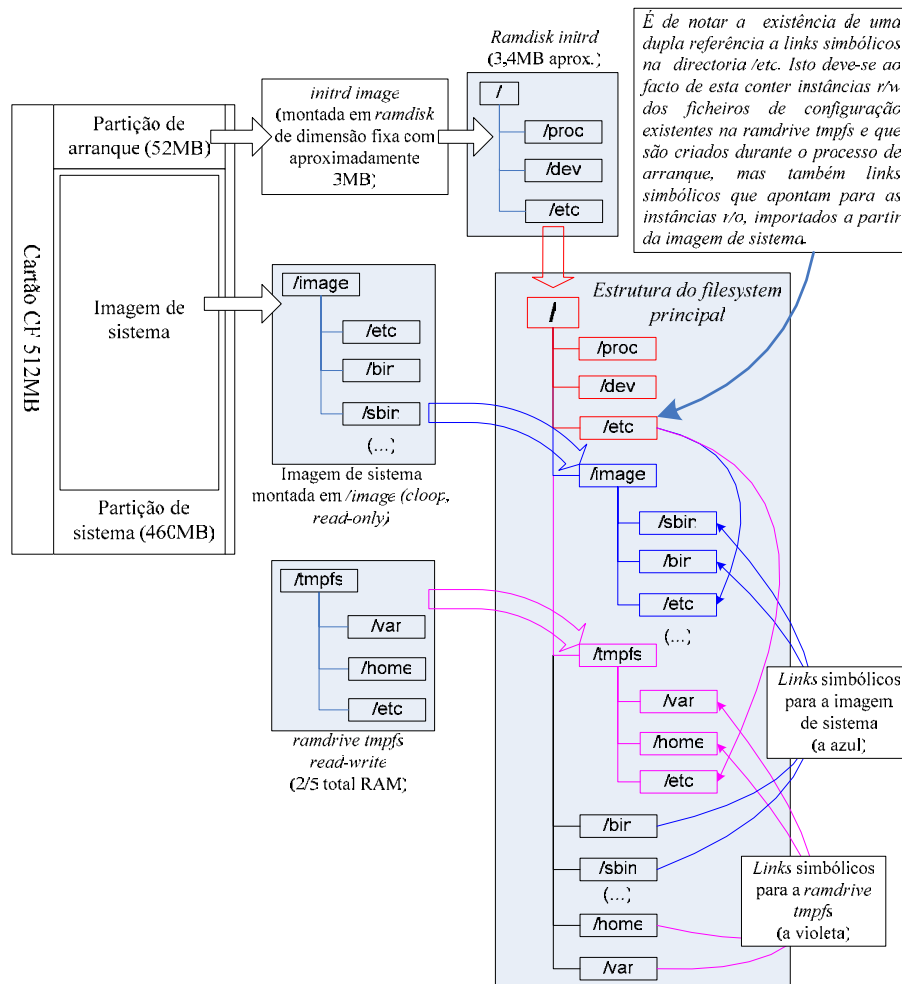



Figura 4.9 –Estrutura de Directorias do Sistema

4.4.4 *Stateless Plug and Play*

A implementação dos mecanismos de *stateless plug and play* (cfr. Secção 4.3) previstos nos requisitos funcionais suplementares obriga a que a detecção do *hardware* do sistema seja (re)efectuada a cada ciclo de *power-up* (salvo excepções como impressoras e *scanners* cujas configurações sejam armazenadas com o perfil do utilizador na sua *home*). Este requisito é simultaneamente uma consequência e uma condição necessária à idempotência da plataforma, visto não se armazenar informação de estado local (o que seria de qualquer modo impossível, visto que o volume principal de sistema é *read-only*).

Os processos de detecção e configuração de *hardware* são baseados num conjunto de *scripts* herdados dos existentes na distribuição *Knoppix*. O processo de detecção, incialização e configuração PnP processa-se pela seguinte ordem (Figura 4.10):

- **Detecção de controladores de dispositivos de armazenamento de massa *bootable*.** O sistema efectua a detecção dos controladores IDE (incluindo os de tipo *host* RAID),

SCSI, USB e *Firewire* e dispositivos de armazenamento de massa conectados a partir dos quais possa proceder à carga e inicialização do ambiente de operação.

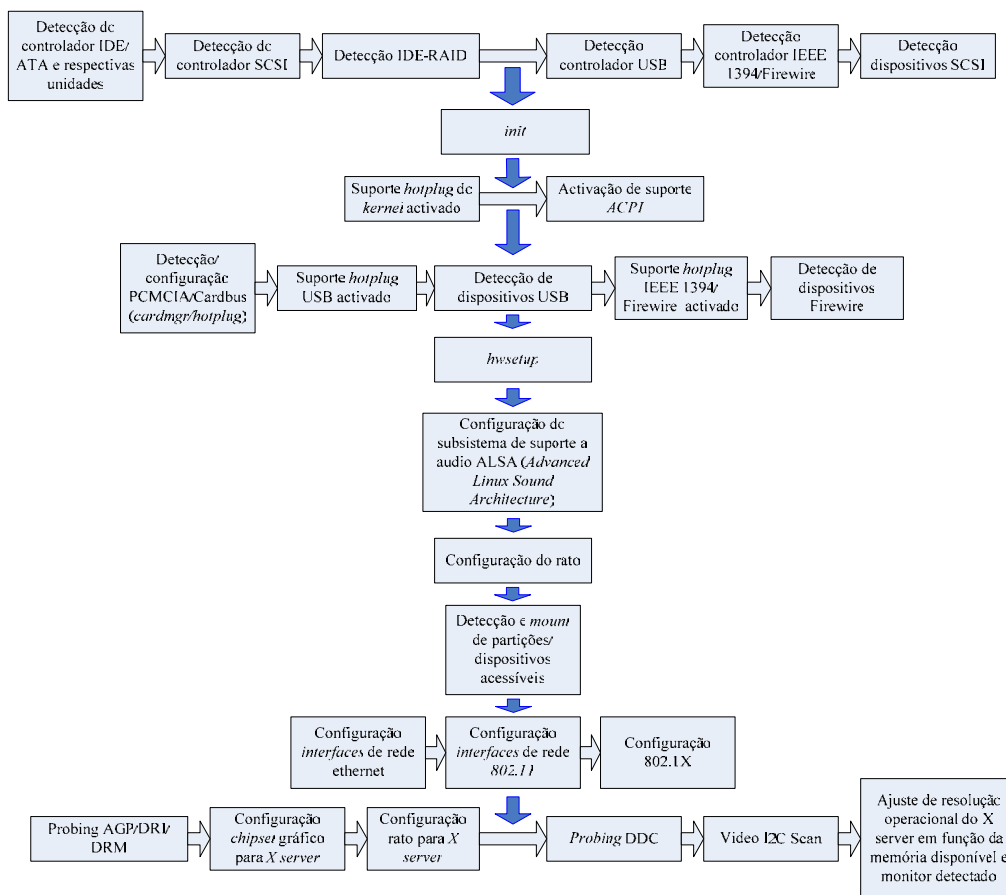



Figura 4.10 – Arranque, Detecção e Configuração PnP

- **Activação dos subsistemas *hotplug* e ACPI do *kernel*.** O subsistema de suporte à remoção e adição de dispositivos *hotplug* é inicializado ao nível do núcleo do sistema operativo. Em seguida é activado o suporte ACPI e são carregados os respectivos módulos para controlo dos botões de sistema (*power-on*, *suspend* e diversos actuadores existentes suportados), gestão de energia (modo ACPI *Standby*, pois o modo *Suspend to RAM* não é devidamente suportado nos *kernels* da série 2.4 [ACPI4Linux]) e bateria, processador (*throttling*) e parâmetros ambientais (como é o caso da ventilação activa).

- **Configuração do suporte PCMCIA/USB/Firewire.** Dada a natureza *hotpluggable* destes dispositivos, a sua configuração é deixada para este estágio do arranque.

  Em primeiro lugar, é activado o suporte PCMCIA/Cardbus, carregando os *drivers* adequados aos controladores existentes no sistema. Em seguida é efectuada a detecção e configuração dos dispositivos conectados nas respectivas ranhuras em articulação com o *Card Manager Service* (*cardmgr*), para dispositivos PCMCIA (16bit, ISA), e com o*, hotplug* para *Cardbus* (PCI, 32bit).

  Em seguida é efectuada a reconfiguração dos subsistemas/controladores USB e *Firewire*, recorrendo ao *hotplug* para a detecção e carga dos *drivers* dos dispositivos encontrados.

- **Execução do *hwsetup*.** Este é um pequeno programa escrito em C que recorre à *libkudzu* da RedHat e a um conjunto de tabelas contendo as correspondências

dispositivos-*drivers* para identificar o *hardware* existente no sistema. Este programa é responsável pela criação dos ficheiros *read/write* da directoria */etc/sysconfig* (que contém a descrição da configuração do sistema) e pelo preenchimento da lista existente em */etc/modules.conf* (com a enumeração dos *drivers* a carregar para o *hardware* detectado). O *hwsetup* encarrega-se ainda de criar a maioria dos *symlinks* adequados na directoria */dev* para os dispositivos detectados.

- **Configuração do sistema de som.** Em Linux existem 2 arquitecturas/APIs para o sistema de som: o *Open Sound System* (OSS) e a *Advanced Linux Sound Architecture* (ALSA), sendo esta última a utilizada pelo IC^3 (a API OSS é suportada via emulação). Neste estágio do arranque, recorrendo ao *script alsa-autoconfig*, o ambiente de operação incializa a camada ALSA, carregando o(s) *driver(s)* apropriados para o *hardware* detectado e a camada de emulação OSS, configurando em seguida os *mixers* para níveis apropriados.
- **Detecção/configuração do rato.** Nesta etapa é levada a cabo a configuração do tipo de rato detectado no sistema (pelo *hwsetup*, declarado em */etc/sysconfig/iccube*), no caso da utilização de um rato USB ou PS/2.
- **Configuração dos *interfaces* de rede.** Após a carga da camada de gestão *cardmgr/hotplug* e da execução do *hwsetup*, o sistema está na posse da informação necessária acerca dos interfaces de rede existentes, procedendo à sua configuração de acordo com a sua natureza (*wired* ou *wireless*). Existem contudo etapas comuns à configuração da placa *ethernet* tradicional e do adaptador *wireless*: a configuração da pilha protocolar, efectuada via DHCP (*Dinamic Host Configuration Protocol*), minimizando a necessidade de armazenamento informação de estado local, e a activação do *supplicant* para autenticação 802.1X EAPOL.
- **Configuração do subsistema gráfico.** Configuração do *chipset* gráfico (detectado pelo *hwsetup* e declarado em */etc/sysconfig/iccube*), detecção do monitor (se possível, através do canal *DDC* (*Display Data Channel VESA*, que permite interrogar o monitor para obter uma estrutura chamada *Extended Display Information Data*, contendo a descrição pormenorizada das capacidades deste) e geração de ficheiros de inicialização do servidor X. O ficheiro de configuração do servidor X é gerado pelo *script mkxf86config* a partir do *parsing* de um ficheiro *read-only,* contendo um "esqueleto" da estrutura necessária, que é modificado com recurso ao *sed* e *awk*, para gerar uma instância na *ramdrive* criada por forma a funcionar para o *hardware* da configuração utilizada.

  Nos casos em que é suportado, efectua-se ainda um *probe* I2C (*bus* interno existente em muitos subsistemas gráficos utilizado para interligação de componentes e passagem de informação de controlo) para detectar qual a porta de vídeo em utilização (caso dos portáteis, adaptadores *multihead* e sistemas com TV-*Out*).

Uma vez completada esta sequência, o sistema encontra-se capaz de utilizar praticamente todo o *hardware* presente, podendo prosseguir para as etapas seguintes de inicialização de serviços (*filesystems*, verificação de *updates*, entre outros), carga do servidor X e autenticação do utilizador.

4.4.5 Sintese: articulação dos mecanismos *tmpfs, cloop* e *stateless PnP*

A operação em modo *read-only* e sem armazenamento local obtém-se graças à articulação entre três recursos – *tmpfs, cloop* e *stateless PnP*. O diagrama da Figura 4.11 sintetiza a forma como os componentes/recursos descritos nas subsecções anteriores se integram entre si, satisfazendo o requisito funcional da ausência de estado local persistente. O esquema evidencia ainda outra particularidade: a adaptação da distribuição de *Linux* para operação em modo *read-only* foi concretizada de forma a minimizar/eliminar a necessidade de alterar ou modificar aspectos críticos e mais complexos da arquitectura Linux. Privilegiando uma aproximação na qual o sistema operativo acede aos recursos e dados no

modo habitual originalmente previsto torna-se mais simples a posterior manutenção da plataforma IC^3 e a integração de posteriores actualizações da distribuição Linux.

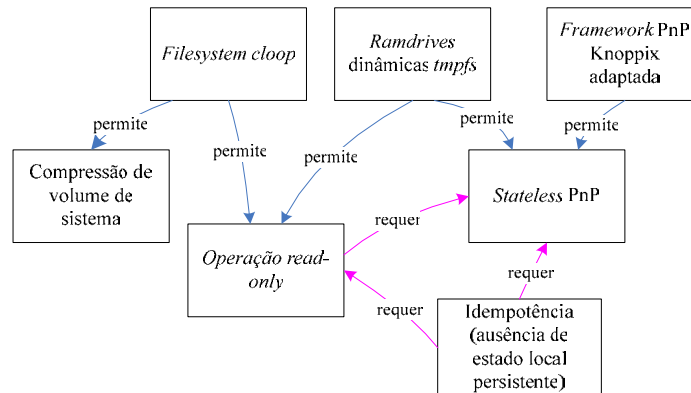



Figura 4.11 – Articulação dos mecanismos *cloop*, *stateless PnP* e *tmpfs*

4.5 Operação do sistema

4.5.1 Actualização/distribuição da Imagem de Sistema

O procedimento de actualização do ambiente de sistema pode ser dividido em três fases distintas:

- Na primeira etapa (**apuramento da existência de versões mais recentes**), o sistema descarrega um ficheiro (IDFILE) a partir do servidor de gestão que contém informações acerca da última versão disponível da imagem contendo o ambiente de operação, seu grau de importância (actualização opcional ou compulsiva) e o seu *hash* MD5. Como curiosidade, note-se que está previsto em futuras versões da plataforma a possibilidade de passar esta informação (excepto o *hash* MD5, por questões de elegância da solução) embebida em *tags* DHCP (DHCP *vendor-options*) recebidas na fase de negociação de IP aquando do arranque.
- Na fase seguinte (**autorização do utilizador**) será dada ao utilizador a possibilidade de decidir se procedimento de actualização é ou não desencadeado. No caso de uma actualização compulsiva o utilizador será simplesmente notificado e o processo será iniciado automaticamente, sem qualquer tipo de intervenção.
- Na fase final (**descarga de nova versão**) o sistema reinicializará e no próximo arranque passará directamente para o *runlevel* 3, a partir do qual inciará o procedimento de descarga da imagem de sistema. Existe um procedimento de autenticação integrado neste passo cujo único objectivo é "abrir" o acesso à rede para permitir a descarga do *update* – em função da política de gestão adoptada este procedimento permite optar por um modelo de operação onde, uma vez criada uma VLAN estanque para manutenção da infraestrutura, todos os clientes autenticados com um determinado ID (*maint*) possam ter acesso aos servidores de actualização e manutenção.

No respeitante ao procedimento de actualização da imagem de sistema, o protótipo IC^3 pode recorrer a duas técnicas distintas, baseadas em protocolos de transferência de ficheiros *unicast* e *multicast*:

- A opção mais conservadora baseia-se na utilização dos protocolos FTP ou SFTP (FTP sobre SSH, sacrificando desempenho em nome da segurança), de natureza *unicast*, sendo cada transferência realizada com recurso a um fluxo de dados ponto-a-ponto.

- Alternativamente, o sistema pode ser configurado para utilizar o UDPCast/*Flamethrower*. O UDPCast [UDPCast] é um sistema baseado em *reliable multicast* desenvolvido para ser utilizado para a replicação de imagens de sistema operativo através de uma rede para instalações em bloco e que mostrou ser adequado, com algumas adaptações, para utilização no contexto da plataforma IC^3. O recurso a este tipo de protocolos deve-se à necessidade de conservar largura de banda nas situações em que a infraestrutura possua um número significativo de estações IC^3 em operação, permitindo aliviar a carga na rede derivada de um processo massivo de actualizações que tanto podem ocorrer por simples coincidência (um número significativo de utilizadores aceita levar a cabo uma actualização opcional) como devido a um processo compulsivo de actualização de todos os postos de trabalho.

No anexo A pode-se encontrar uma explicação mais detalhada sobre o modo como estes protocolos operam e qual o raciocínio subjacente às escolhas efectuadas.

4.5.2 Criação e Manutenção das Imagens de Sistema

O processo de criação e manutenção das imagens de sistema foi desenhado de modo a favorecer a implementação de uma metodologia de gestão de parque informático que torne imperativa a adopção de um conjunto de boas práticas. A metodologia sugerida é fortemente inspirada no modelo em cascata utilizado em processos de engenharia de *software* e contempla os passos ilustrados na Figura 4.12 (cada recuo corresponde a um refinamento/correcção).

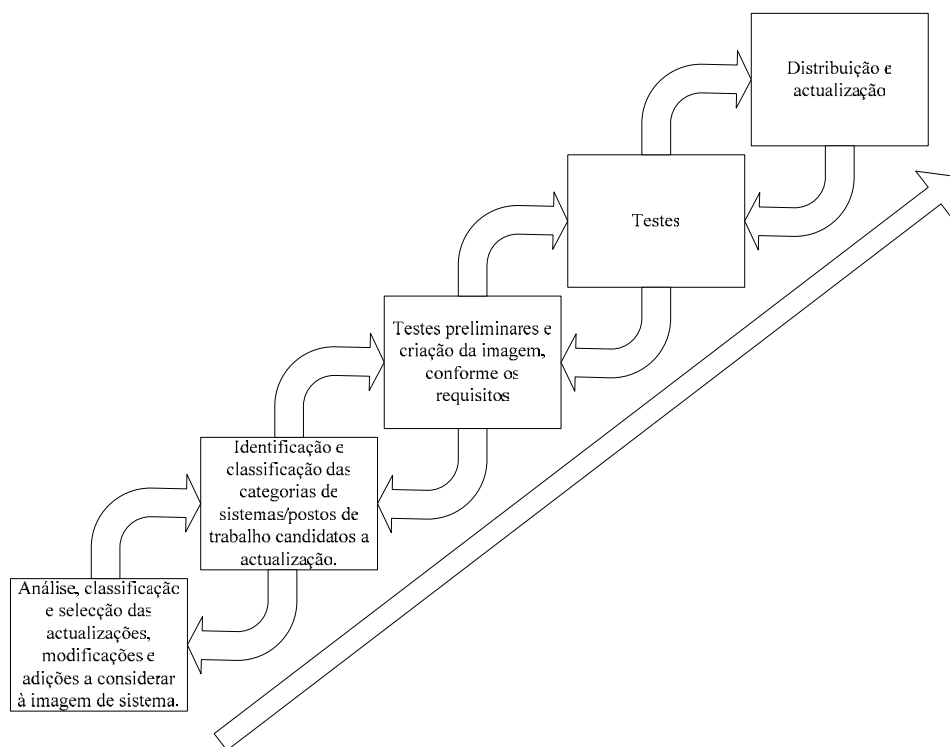



Figura 4.12 – Metodologia de Criação e Manutenção de Imagens de Sistema

Sendo impossível instalar/adicionar directamente um *update* ou um componente na imagem de sistema sem criar uma imagem de sistema totalmente nova, torna-se necessário um processo bem definido antes de efectuar qualquer alteração. O processo é simples, se bem que potencialmente demorado, e obriga implicitamente a testes antes que a imagem seja colocada *on-line* (Figura 4.13).

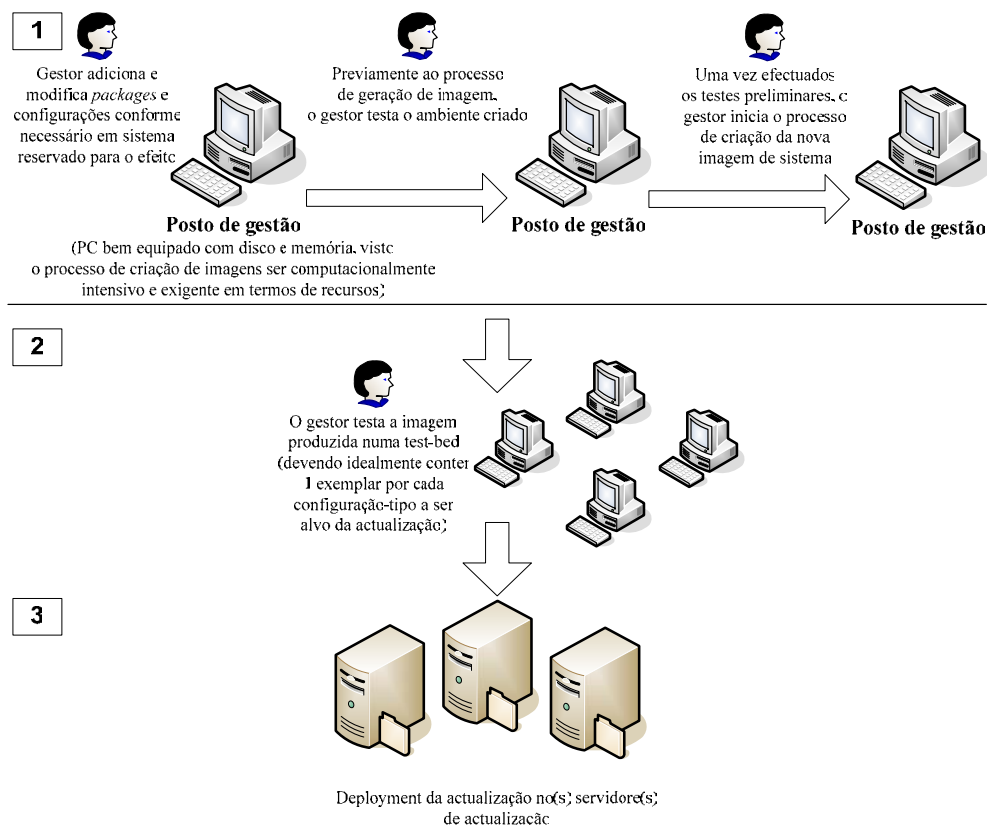



Figura 4.13 – Processo de Criação de Imagens de Sistema Operativo

Tal é possível porque a imagem é criada em modo *offline* num sistema reservado para o efeito, sendo possível testar as alterações à medida que estas são efectuadas (graças ao uso de um ambiente *chroot*) e antes de iniciar a criação da imagem. Para possibilitar o teste preliminar de alterações mais complexas, que envolvam modificações ao processo de arranque é ainda possível utilizar o QEMU [QEMU], para testar integralmente a imagem criada dentro de uma máquina virtual simulada para o efeito em tempo real.

A adição e remoção de *packages* e componentes de *software* é processada utilizando as ferramentas de gestão herdadas da distribuição *Debian Linux* (*dpkg-tools*), que permitem levar a cabo estes procedimentos de forma simples e segura – desde o simples processamento de dependências até actualizações de fundo com grande complexidade que envolvam um grande número de componentes envolvidos.

Uma vez criada a imagem, o gestor deverá conduzir um conjunto de testes em condições semelhantes às dos postos de trabalho onde esta será instalada. Para o efeito a *testbed* a utilizar deverá contemplar, idealmente, um exemplar de cada configuração-tipo a ser abrangida pelo processo de actualização.

Apenas após o teste do ambiente gerado este deverá ser disponibilizado nos sevidores, assinalando-se no respectivo ficheiro (ficheiro IDFILE, conforme Secção 4.5.1) o grau de prioridade e quais os sistemas a que se destina.

4.5.3 Autenticação, autorização e *accounting* dos utilizadores (AAA)

Tendo em vista a uniformidade de procedimentos de autenticação entre os modos de operação *wired* e *wireless*, optou-se por suportar o protocolo 802.1X do IEEE, integrado com um *backend* RADIUS que é suportado, por sua vez, por um serviço de directoria LDAP. Devido às fragilidades do protocolo WEP (*Wired Equivalent Privacy*) [iLabs 2002], à falta de

uniformidade do suporte WPA (*Wi-Fi Protected Access* [Wi-Fi 2003]) entre implementações distintas e ao atraso da adopção do *standard* 802.11i/WPA2 [IEEE 2004], a *framework* 802.1X surge como uma alternativa natural devido a um conjunto de características atractivas:

- **Independência do meio físico.** O 802.1X foi concebido para operar independentemente do meio físico utilizado pela rede. Esta característica é importante uma vez que permite viabilizar o requisito funcional da autenticação uniforme (cfr. Secção 4.1).
- **Criação de perímetro seguro.** O uso de 802.1X permite criar um perímetro seguro que limita o acesso à infraestrutura de rede em função do utilizador conectado estar ou não autorizado. Tal é possível porque se incorpora o suporte para *port-based authentication*, permitindo a autenticação e acesso por ponto de rede.
- **Autenticação em canal seguro.** Sendo baseado numa adaptação do protocolo EAP (*Extensible Authentication Protocol* [RFC3748]), originalmente especificado no RFC2284 e actualizado mais tarde no RFC3748, o 802.1X suporta uma diversidade de subprotocolos para suportar a transacção dos dados de autenticação.

A Figura 4.14 ilustra a operação do 802.1X no contexto do projecto IC^3. Cada posto de trabalho – denominado suplicante, na nomenclatura 802.1X – transporta a informação EAP (*cfr. caixa na página seguinte*) directamente sobre a tecnologia de (W)LAN em uso. Esta solução, designada EAPOL (*EAP over LAN* [IEEE 2001]), suporta *Token-Ring*, *Ethernet* ou quaisquer derivados que utilizem o mesmo formato base de pacote.

Entre o suplicante e as *Port Authentication Entities* (PAEs, como é o caso dos *access point 802.11,* do *switch Ethernet* ou da *bridge hostapd* da Figura 4.11) a troca de informação processa-se via EAPOL. Cada PAE encarrega-se de operar como um intermediário no processo de autenticação 802.1X, processando a troca de informação ao nível da camada de ligação sem armazenar qualquer informação do utilizador ou suas credenciais. Cada porta, seja ela virtual (no caso de um AP ou *bridge hostapd*) ou física (no caso de um *switch*) pode encontrar-se num estado autorizado (deixando passar tráfego para o interior da rede da organização) ou não-autorizado, conforme o estado do processo de autenticação do utilizador que a ela se encontra conectado.

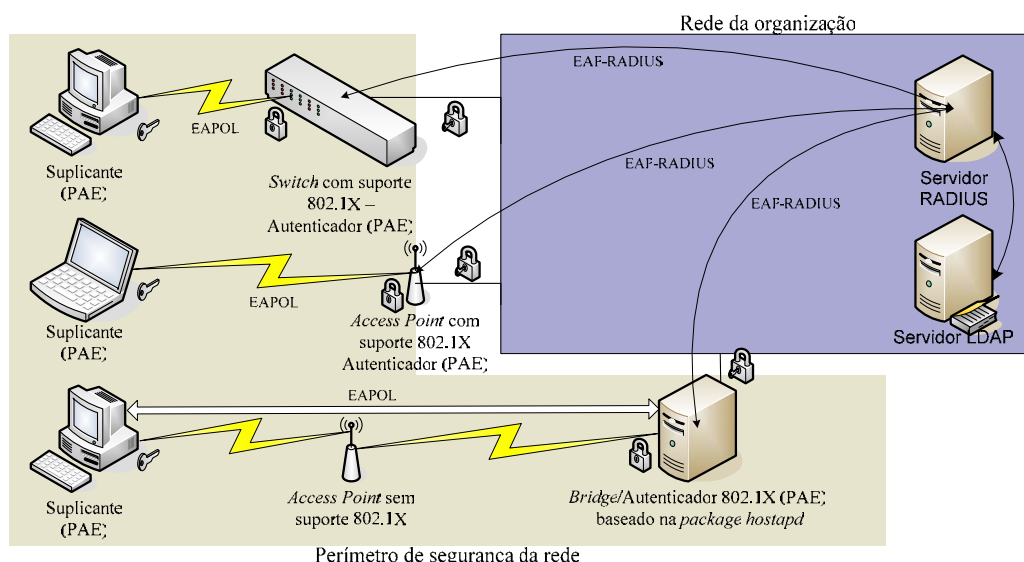



Figura 4.14 – Operação do 802.1X no Contexto do IC^3

O Protocolo EAP

Este protocolo foi originalmente introduzido no início da década de 90 para suporte de autenticação em ligações PPP (*Point-to-Point Protocol*) [RFC1661], tendo sido concebido de modo a poder operar a nível da camada de ligação, de forma independente do meio físico. O formato do pacote EAP é o seguinte:

Header (PPP ou LAN)	Código (1)	Identificador (1)	Tamanho (2)	Dados (tamanho variável)	
				Tipo (1)	Dados associados ao tipo (campo com tamanho variável)

- O código identifica o tipo de pacote EAP, sendo utilizado para interpretar o campo de dados e permitindo identificar as operações em curso (códigos *request, response, success, failure,* respectivamente).
- O campo *identificador* contem um inteiro (*unsigned*) que é utilizado para identificar a sequência das operações, permititndo fazer a associação entre pedidos e respostas.
- O campo *tamanho* contém o número de *bytes* no pacote inteiro, incluindo os campos *código, identificador, tamanho* e *dados*. Em alguns protocolos da camada de ligação lógica pode ser necessário perfazer um determinado tamanho, recorrendo para isso ao *padding.* O EAP assume que os dados em excesso do campo *tamanho* são *padding*, ignorando-os.
- O campo de *dados* tem dimensão variável e, dependendo do tipo de pacote, pode mesmo ter dimensão zero.

A autenticação no EAP é delegada a um subprotocolo ou método, fazendo assim jus à sua natureza extensível (como o próprio nome indica). Quando os métodos existentes não dão resposta a um conjunto específico de novas necessidades torna-se assim possível preencher a lacuna criando novas extensões.

A figura mostra uma transacção EAP simples. Os métodos ilustrados não são geralmente recomendados para ambientes *wireless*, mas são adequados como exemplo de operação. O formato das transacções é do tipo *Request/Método* e *Response/Método*.

- Numa primeira fase, o autenticador requer ao cliente que se identifique. O cliente solicita ao utilizador a sua identificação, recolhendo as devidas credenciais (nome do utilizador) que devolve na respectiva resposta.
- Com o utilizador identificado, o autenticador gera um *authentication challenge*, requerendo ao utilizador que se autentique utilizando um método semelhante ao CHAP com *hash codes* MD5. O utilizador responde com um NAK, sugerindo que a autenticação se processe via *token card* (*generic token card,* como é o caso dos cartões do tipo RSA SecurID).
- O autenticador requer um *token card challenge*, solicitando a sequência numérica do cartão. O utilizador responde, sendo os dados enviados no pactote de resposta.
- Se a resposta do utilizador não for correcta, a autenticação não é possível, sendo feita uma nova tentativa com novo par *Request/Response*. Nesta nova tentativa, o processo é bem sucedido.
- Uma vez que a autenticação foi bem sucedida, o autenticador emite uma mensagem assinalando o sucesso da operação.

Entre as PAEs e o servidor de autenticação (*backend*) a informação EAP é transportada em pacotes RADIUS [RFC2865]. O suplicante (que nesta fase não necessita ainda de um endereço IP) efectua uma negociação EAP com o servidor de autenticação, mediada pelo autenticador através de uma porta inicialmente não-autorizada. Ao recorrer ao protocolo RADIUS beneficia-se de uma enorme flexibilidade no respeitante ao suporte para a base de dados dos utilizadores, optando-se no caso do IC³ por um servidor de directoria LDAP central. Na prática o servidor RADIUS opera como uma *gateway* para o servidor de directoria central, cujos *logs* de operação fornecem informação de autenticação e autorização dos utilizadores, assim como dados adicionais de *accounting* das sessões – acrescentando assim a desejável garantia de não-repudiação.

Na configuração adoptada são suportados os métodos EAP-TTLS e PEAP, de modo a permitir que a autenticação se processe de forma segura utilizando o protocolo MSCHAPv2 (que, *per se*, não é um método criptograficamente protegido). Tanto o EAP-TTLS como o PEAP operam de forma semelhante, estabelecendo numa primeira fase ("autenticação externa") um tunel TLS (*Transport Layer Security*) encriptado recorrendo a certificados digitais armazenados no servidor de autenticação para validar se a rede é confiável antes de prosseguir. Na segunda fase é efectuada a autenticação dita "interna" (chama-se assim, visto operar sob a protecção do tunel TLS previamente estabelecido) utilizando para o efeito o protocolo EAP-MSCHAPv2 (Figura 4.15).

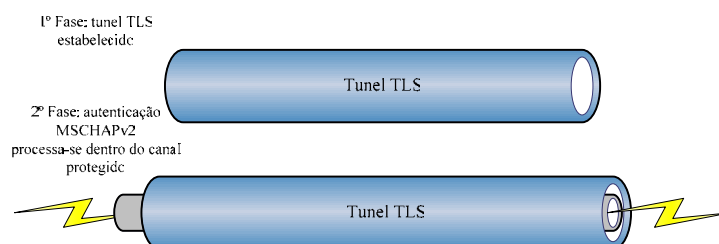



Figura 4.15 – Autenticação MSCHAPv2

A diferença entre o PEAP e o EAP-TTLS reside na forma como a autenticação interna é processada. No caso do EAP-TTLS pode recorre-se a uma simples troca de pares atributo-valor dentro do túnel encriptado, permitindo assim o recurso a métodos de autenticação internos que não sejam baseados em EAP (caso do CHAP ou PAP), enquanto que no caso do PEAP o canal protegido é utilizado para iniciar um segundo procedimento EAP.

A operação do 802.1X em ambientes *wireless* (*cfr. caixa na página seguinte*) acrescenta ainda dois benefícios suplementares:

- **Protecção contra ataques do tipo *man-in-the-middle*.** Com a relativa acessibilidade do preço dos APs, torna-se possível a um atacante introduzir numa rede um ponto de acesso comprometido (*rogue AP*) com o objectivo de intersectar a informação circulante. O 802.1X apenas permite o acesso à rede quando se verifica que o AP/Autenticador faz parte da desta e o utilizador está devidamente autenticado e autorizado , gerando para o efeito um par únivoco de chaves de encripação válidas para a sessão estabelecida.
- **Chaves WEP dinâmicas.** O 802.1X, quando utilizado em redes *wireless,* permite ultrapassar a principal fragilidade do WEP configurado com chaves manualmente definidas, a qual se prende essencialmente com a imutabilidade destas ao longo do tempo e que as torna vulneráveis. As chaves WEP são automaticamente criadas quando a autenticação é bem-sucedida sendo possível fazer a rotação dinâmica destas via mensagens EAPOL.

Contudo, há que salientar o facto de o protocolo 802.1X ter algumas fragilidades, nomeadamente no que diz respeito a redes *wired*. Uma vez autenticado o utilizador aquando da conexão inicial, a porta onde este está conectado é colocada em estado autorizado e o restante tráfego de rede pode passar sem qualquer tipo de protecção. Isto implica que um atacante possa, por exemplo, inflitrar um *hub* entre o posto de trabalho e o ponto de rede de modo a cruzar o perímetro seguro (utilizando um segundo computador ligado ao *hub*). Mesmo que o computador da vítima seja desconectado apenas por breves instantes, para colocação do *hub*, ele irá reautenticar a ligação ao *switch* mal a ligação seja reestabelecida. De seguida o atacante apenas tem de reconfigurar o seu *shadow* PC com o mesmo MAC *address* e endereço IP da vítima. Como o 802.1X não prevê autenticação *follow-per-packet* (uma vez que só autentica a conexão) o atacante encontra-se livre para utilizar protocolos *stateless* (ICMP, UDP) para concretizar as suas intenções – note-se que o uso do TCP se encontra vedado, pois o computador vítima (que tem os mesmos endereços MAC e IP do *shadow computer*) fará o *reset* de todas as conexões iniciadas pelo *shadow host* (Figura 4.16).

802.1X em ligações *wireless*

1- O suplicante associa-se à rede 802.11.
2- A negociação 802.1X é iniciada com o envio de uma mensagem *EAPOL-Start* (opcional). Em resposta, incia-se uma negociação EAP *standard*, começando o autenticador por enviar uma mensagem *Request/Identity* à qual o suplicante responde com uma mensagem *Response/Identity* que é passada ao servidor RADIUS como um pacote *Radius-Access-Request*.
3- O servidor RADIUS determina qual o tipo de autenticação requerida e envia uma mensagem *EAP-Request* (ou *EAP-Request/PEAP*, se se utilizar PEAP), encapsulada num pacote *Radius-Access-Challenge* solicitando o método a utilizar que, quando atinge o AP, é transmitida ao suplicante como uma mensagem *EAP-Request/Method*.
4- O suplicante solicita ao utilizador as suas credenciais e envia uma mensagem *EAP-Response* que é encapsulada pelo AP numa mensagem *Radius-Access-Request*. De seguida estes procedimentos são repetidos quantas vezes for necessário para completar o processo de autenticação – o que poderá implicar multiplos passos se for requerida uma troca de certificados.
5- O servidor RADIUS concede o acesso, enviando um pacote *Radius-Access-Accept* que é traduzido pelo AP numa mensagem *EAP-Sucess*.
6- O AP ditribui as chaves WEP ao suplicante utilizando uma ou mais mensagens *EAPOL-Key*. Uma vez configurado o suplicante com as chaves, pode ser iniciada a transferência de dados.
7- Quando o suplicante terminou a sessão é emitida uma mensagem *EAPOL-Logoff* para que a porta seja colocada novamente em modo não-autorizado.

Figura 4.16 – Intrusão em Redes *Wired* com 802.1X

Numa rede corporativa o risco derivado desta vulnerabilidade vai depender em grande parte da política de segurança adoptada. Se se considerar necessário que a conexão seja realmente segura, a melhor solução passará pela adopção do protocolo IPSEC [RFC4301] cujas características permitem efectuar a autenticação da conexão e do tráfego circulante (autenticação *follow-per-packet*). De momento esta tecnologia não é suportada na plataforma IC^3, sendo no entanto passível de incorporação futura.

Outra opção passa pelo recurso aos protocolos LEAP (*Lightweight EAP*, também conhecido como *EAP-CiscoWireless*) ou PPTP (*Point-to-Point Tunneling Protocol*). Estes dois protocolos operam criando túneis (alegadamente) seguros a nível da camada 2 do modelo ISO, recorrendo a autenticação baseada no protocolo MSCHAP (MSCHAPv1 no LEAP e PPTPv1, e MSCHAPv2 no PPTPv2, sendo ambas as alternativas inerentemente inseguras se a transacção dos dados não for protegida por outro método) e a algoritmos de encriptação de canal como o MPPE-40/56/128 (*Microsoft Point-to-Point Tunneling Protocol* com chaves de *40/56/128 bit* utilizado no PPTP e que é baseado no protocolo RSA RC4) e que foram difundidos em ambientes *wireless* para colmatar as fragilidades do WEP.

Estes protocolos são reconhecidamente inseguros (o modo de operação no respeitante ao processo de autenticação dos utilizadores é demasiado semelhante em ambos os casos), existindo mesmo uma ferramenta – o *asleap* [Wright 2004] que permite interceptar as credenciais dos utilizadores (Figura 4.17). Contudo, deve ser feita uma ressalva ao caso do PPTP com autenticação EAP-TLS (suportada no PPTP v2), que não padece dos problemas decorrentes do uso do MSCHAP [Schneier 1999], embora padeça das restantes fragilidades estruturais derivadas da concepção do protocolo [Robinson 2002], nomeadamente a vulnerabilidade a ataques DoS, devido ao facto de os dados necessários ao estabelecimento das conexões e sua manutenção circularem em *cleartext*.

```
asleap 1.4 - actively recover LEAP/PPTP passwords. <jwright@hasborg.com>
Using the passive attack method.

Captured PPTP exchange information:
        username:           scott
        auth challenge:     e3a5d0775370bda51e16219a06b0278f
        peer challenge:     84c4b33e00d9231645598acf91c38480
        peer response:      565fe2492fd5fb88edaec934c00d282c046227406c31609b
        challenge:          7c00a1a403ca7df5
        hash bytes:         816b
        NT hash:            18073343f630b5f82c38c03437f2816b
        password:           turquoise
```

Figura 4.17 – Intercepção de Credenciais com o *asleap*

4.5.4 Suporte para Sistemas de Ficheiros de Rede

Nas versões *wired* da plataforma IC^3 o acesso aos dados de utilizador e à sua *roaming home* é efectuado via rede, recorrendo para esse efeito a *network filesystems* como o NFSv3 [RFC1813] ou o SMB [Sharpe 2002]. Esta aproximação tem como principais vantagens a consolidação das necessidades de armazenamento num ponto central e a ausência de estado local nos postos de trabalho.

O NFSv3 possui as características de fiabilidade e maturidade que o tornam conveniente para o fim desejado, além de suportar funcionalidades vitais quando se lida com sistemas de ficheiros "tipo-UNIX", tais como o suporte de *links* simbólicos. Na arquitectura do IC^3 prevê-se a existência de um servidor NFS, tipicamente apoiado no *backend* por um dispositivo NAS (*Network Attached Storage*) ou por uma SAN (*Storage Area Network*), tendo em vista a consolidação do armazenamento.

Assim sendo, uma vez processada a fase inicial de autenticação 802.1X o sistema prossegue com o acesso (*mount*) à *roaming home* NFS. As credenciais (nome de utilizador e *password*)

são reutilizadas para autenticação LDAP (processo efectuado através de uma ligação TLS) e acesso aos *automount maps* (armazenados no servidor LDAP), que são utilizados pelo serviço *automount* para activar automaticamente o acesso às *roaming homes* dos utilizadores (Figura 4.18).

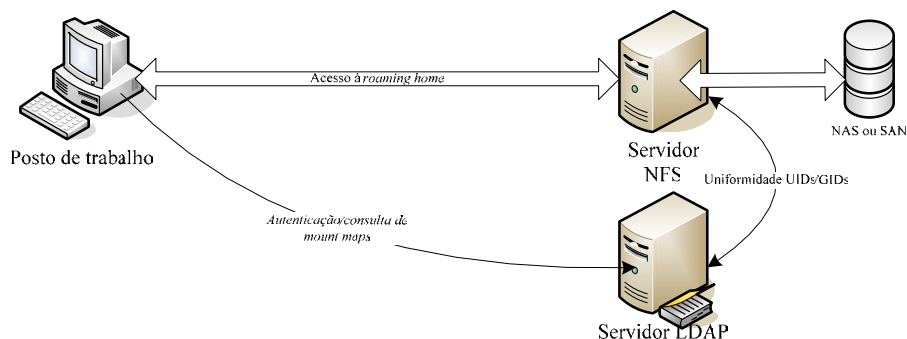



Figura 4.18 – Acesso ao Sistema de Ficheiros Remoto

Para este efeito criou-se a nível da directoria LDAP uma nova *OU* (*Organizational Unit*) chamada *autofs*, que se encontra estruturada conforme ilustra a Figura 4.19. Adicionalmente, e a bem da interoperabilidade entre plataformas (conforme o requisito funcional da interoperabilidade previamente identificado), acrescentou-se suporte para o protocolo SMB, tendo em vista o acesso a *fileservers* e partilhas de rede existentes em PCs ou servidores com sistemas operativos da família *Windows*. O acesso a estes recursos pode ser automatizado de forma análoga ao que é feito para as *roaming homes*, bastando para o efeito configurar os mapas do *automounter* na directoria LDAP.

No entanto, o NFS bem como o SMB sofrem de uma fragilidade derivada da impossibilidade de replicação dos *fileservers* que pode comprometer de forma crítica a disponibilidade e/ou a escalabilidade da arquitectura. Nesta situação o desenvolvimento de uma solução de *clustering* sobre NFS pode ocorrer como uma alternativa, recorrendo a um *load balancer* como o LVS (*Linux Virtual Server*) ou a um sistema de DNS *round-robbin* que permitisse operar de acordo com os modos ilustrados na Figura 4.20.

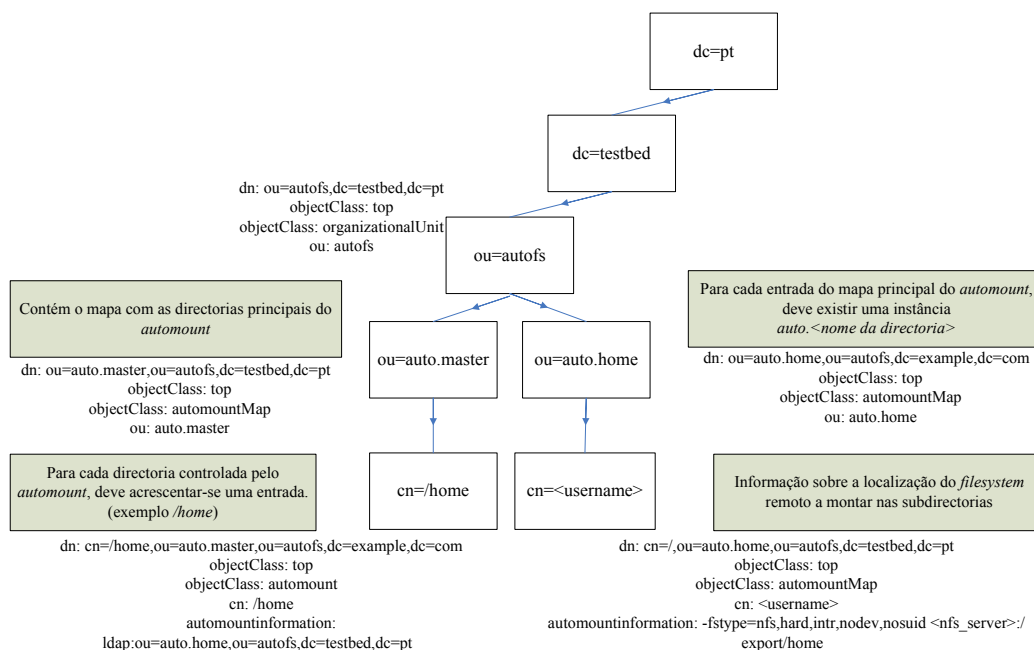



Figura 4.19 – Estrutura da *autofs*

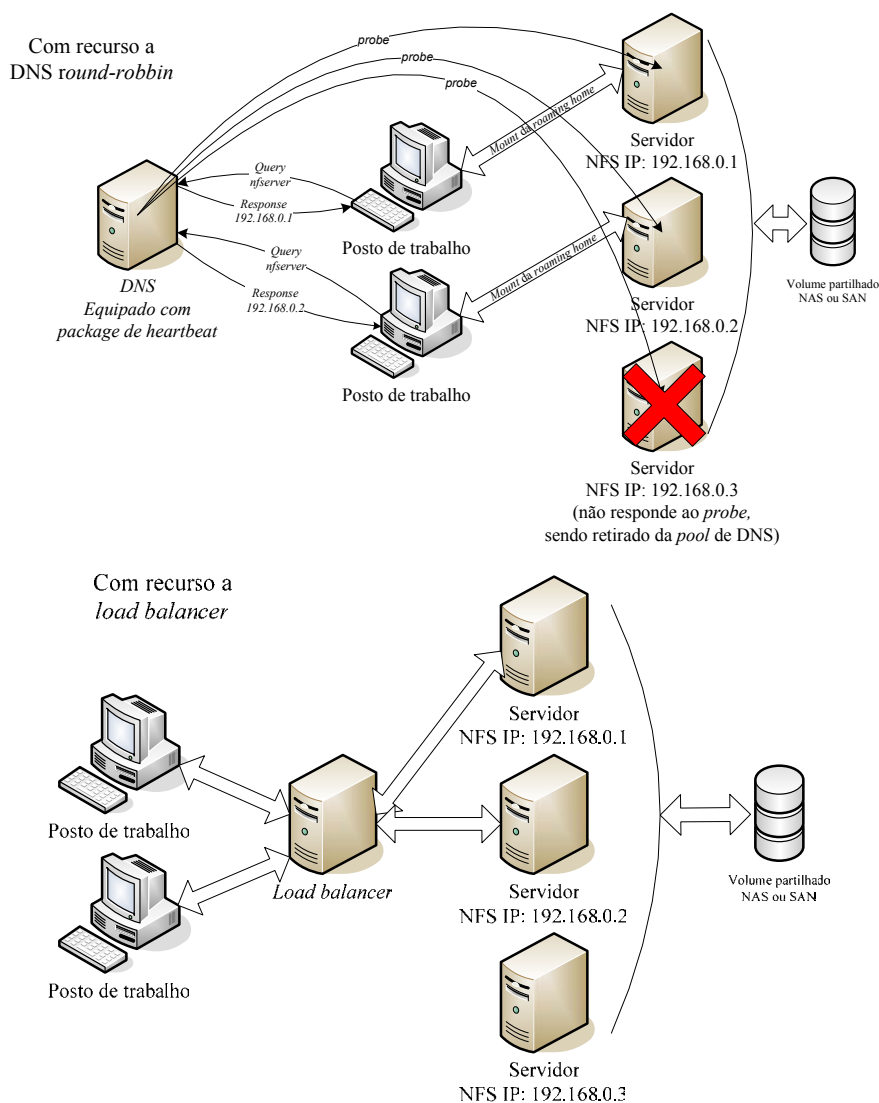



Figura 4.20 – Potenciais Soluções de *Clustering* sobre NFS

Como o NFS é implementado com recurso a operações atómicas *stateless* (sobre chamadas *Sun RPC*) seria possível em teoria concretizar um cenário deste género não obstante um conjunto de problemas, nomeadamente ao nível dos *locks* e dos mecanismos de *caching*.

Cada servidor NFS mantém o seu próprio sistema de gestão de *locks* (que, no caso do NFSv3, são atómicos, ao contrário do que sucedia nas versões anteriores) de modo a manter a consistência quando ocorrem acessos concorrentes a um mesmo ficheiro. Para operar de forma coerente, as soluções teriam de garantir os seguintes requisitos:

- Os acessos de escrita e leitura a um determinado ficheiro nunca poderiam ser efectuados por mais de um servidor.
- Os *locks* deveriam ser mantidos de forma consistente entre servidores distintos e não apenas ao nível local de cada instância.

A existência de mecanismos de *caching* coloca também problemas, nomeadamente em termos de coerência no acesso aos dados:

- Ao nível do servidor, devido ao recurso a técnicas de *delayed write* que permitem que dados alterados por uma operação de escrita numa dada página de memória (utilizada para *buffer* de dados do *filesystem*) apenas sejam escritos para disco quando a página for necessária para outro fim ou de 30 em 30 segundos, através da operação *sync*.

- Ao nível do cliente, devido à existência de mecanismos de *caching* a nível dos clientes que permitem que possam potencialmente existir versões distintas do mesmo ficheiro ou porções destes nos diversos sistemas cliente, dado que as escritas no servidor podem não resultar na actualização imediata das cópias dos dados existentes nas *caches* locais. Assim, os clientes são obrigados a validar os dados em *cache* antes do seu uso, recorrendo a um mecanismo de *polling* com uso de *timestamps* para garantir que, se os dados no servidor tiverem sido alterados em data posterior aos existentes em *cache* local, esta seja invalidada e seja feito um novo pedido ao servidor para envio dos dados actualizados. De acordo com a implementação NFS podem ainda ser suportados mecanismos de *delayed write* e *read ahead* (com recurso a serviços de *block input/output* que operam em paralelo com as operações NFS, permitindo a execução de leituras de blocos em avanço dos requisitados pelas operações de I/O e escritas não bloqueantes, fazendo o *flush* dos blocos em paralelo).

  A semântica das operações sobre as quais estes mecanismos foram implementados foi pensada tendo apenas em conta a situação em que existe um servidor para vários clientes NFS e apenas permitem a replicação segura de servidores com volumes *read-only* partilhados (com os clientes a acederem de forma persistente a um mesmo servidor). No caso da existência de vários servidores com acesso simultâneo de leitura e escrita, cada um com as suas *caches* locais, que partilham um mesmo volume de dados (sujeito a operações de leitura e escrita) e de vários clientes cujas respectivas *caches* são actualizadas e mantidas em sincronismo de contexto com os servidores acedidos surge um sério problema de coerência e consistência, provocado pelas várias situações resultantes das combinações de um vasto leque de possibilidades (como aquelas em que um cliente em operações sucessivas de I/O acede a servidores distintos, ocorrências de falhas em operações de I/O, entre outros) que ambas as arquitecturas (DNS *round-robbin* e *load-balancer*) propostas preconizam – e que invalidam de todo esta possibilidade. A possibilidade da existência de mecanismos *de* DNS *caching* nos clientes também coloca um conjunto de problemas em termos de disponibilidade e *failover* (um servidor indisponível e removido da *pool* de DNS pode continuar a ser resolvido pela *cache* local, podendo também suceder o recíproco).

- A potencial ausência de uniformidade de *filehandles* entre servidores distintos, para um mesmo ficheiro, coloca também problemas. Ambas as soluções oferecem uma solução viável para os casos em que a de um dado servidor ocorre antes do acesso do posto de trabalho ao volume. Contudo, se a indisponibilidade se der durante a operação (e não na fase preliminar ao acesso), a forma como as soluções apresentadas se comportam pode não garantir *failover* transparente.

  A inexistência de uniformidade dos *filehandles* entre dois servidores NFS com cópias dum mesmo *filesystem* partilhado (para um mesmo ficheiro replicado em dois servidores, os *filehandles* serão forçosamente distintos) impossibilita a comutação transparente entre instâncias: em caso de falha, os clientes retomam a operação com base em *filehandles* inválidos (*stale*). O mesmo não sucede se os servidores em questão partilharem um mesmo volume (*cfr. caixa*). A adopção de um volume partilhado não resolve o problema da impossibilidade da replicação de servidores com acesso de escrita e leitura a operar em paralelo (estando assim ambas as soluções propostas descartadas) mas abre a porta para a implementação de soluções de alta disponibilidade baseadas em mecanismos de *failover*, com um servidor a operar de cada vez e com um ou mais *backups* prontos a tomar o seu lugar em caso de falha.

Os *filehandles* NFS e a Possibilidade de *Failover*

De acordo com o protocolo NFS, para cada ficheiro é gerado um *filehandle* com o identificador do respectivo *filesystem*, o *i-node* do ficheiro e um número de sequência adicionado para impedir confusão se o ficheiro for apagado, e o *i-node* reutilizado (alguns autores [Couloris 2001] afirmam que este valor, também chamado de *generation number*, é armazenado junto de cada ficheiro por intermédio da camada VFS (*Virtual Filesystem Layer*) e modificado a cada reutilização do *i-node* por simples incremento de um contador, sendo utilizado pelo *daemon* NFS para o *filehandle* mas, de facto, nem todas as implementações se comportam do mesmo modo).

Identificador do filesystem	i-node do ficheiro	Número de sequência do *i-node*

A estrutura de um *filehandle* NFS

Esta forma de gerar os *filehandles* impede que dois servidores com cópias de um mesmo *filesystem* possam ser utilizados numa solução de *failover*, pois para um mesmo ficheiro a possibilidade de os *filehandles* gerados serem idênticos é quase nula: graças à estrutura do *filehandle* NFS, a simples diferença entre os *i-nodes* pode provocar este problema, visto estar-se a lidar com cópias dos *filsystems*, não com réplicas exactas.
Contudo, se todos os servidores NFS acederem a um mesmo volume e/ou dispositivo (um dispositivo NAS iSCSI ou *fibre channel n-ported*, por exemplo) é possível uniformizar os componentes dependentes da geometria do dispositivo (com algum cuidado, garantindo que o dispositivo é o mesmo em todos os servidores), o *i-node* e o número de sequência. Assim sendo, uma configuração de *failover* com armazenamento partilhado é, em teoria, possível. De facto, é precisamente assim que funcionam as soluções de cluster de alta disponibilidade (HA, *High Availability*) para *NFS* comercializadas pela *Sun Microsystems*.
Por simples experimentação foi possível constatar que, de facto, numa configuração HA simples com dois servidores (configuração "activo-passivo", baseada na manutenção de um servidor em *standby* que toma o lugar do principal em caso de falha, usando para o efeito sistemas Linux com *kernel* 2.6.9 e *kernel-mode* NFS) é possível que os clientes migrem de forma transparente da instância primária para a secundária quando é simulada uma falha. Não foram detectadas perdas de coerência, tendo apenas sido registada uma paragem momentânea de alguns segundos, nos clientes, após a redisponibilização do serviço.
No entanto, ainda que seja potencialmente possível desenhar mecanismos efectivos de *failover* para NFS, considerou-se que o grau de incerteza envolvido e o esforço inerente ultrapassavam o âmbito do IC^3.

Estes factores condicionam de forma determinante a possibilidade de desenvolver uma solução simples de *load-balancing* para NFS, conduzindo a outras soluções.

Para a questão do *failover,* ainda que existam produtos comerciais como o *Lifekeeper* [Steeleye] e o *RedHat Cluster Manager* [RedHat 2002], a solução poderia ser obtida usando uma arquitectura HA "activo-passivo" e ferramentas gratuitas como o *heartbeat* [Heartbeat] e o DRDB [LINBIT]. Fica no entanto por resolver a questão da escalabilidade.

Em alternativa, pode optar-se por um *filesystem* de filosofia puramente distribuída, tal como o Lustre [Lustre], o GFS [RedHat 2004], o *v9fs* (um port do *Plan 9 file protocol* [v9fs]) ou o AFS [Campbell 1998], mas tal implicaria estudos de viabilidade e adequação que terão de ficar adiados para futuros desenvolvimentos do IC^3. Outra hipótese interessante a considerar, a médio/longo prazo, será a migração para NFSv4 [RFC3530], uma vez amadurecido e estabilizado o protocolo e a respectiva implementação para Linux. Seria assim possível beneficiar de novas funcionalidades, tais como suporte a replicação de servidores, *namespace* global (que permite transparência de acessos) e mecanismos de migração de *locks* (resolvendo assim o problema derivado da falta de uniformidade entre servidores distintos).

4.5.5 Acesso a *Display* Remoto: o Protocolo NX

Na perspectiva do paradigma de *balanced computing*, tal como apresentado e descrito na Secção 3.3, o suporte para protocolos de *display remoto* (X11, RDP, VNC) é condição vital. A solução tradicional passa por integrar o suporte para os clientes de cada protocolo específico no *desktop*, embora o ideal fosse dispor de um modo uniforme e independente do protocolo utilizado para aceder aos diversos tipos e sistemas – um cliente universal.

Além disso, cada protocolo tem as suas fragilidades. O *X/Windows*, por exemplo, é bastante eficiente no uso da largura de banda disponível (pouco *overhead*) mas padece de problemas provocados pelo excesso de *round-trips* entre o cliente e o servidor (Figura 4.21).

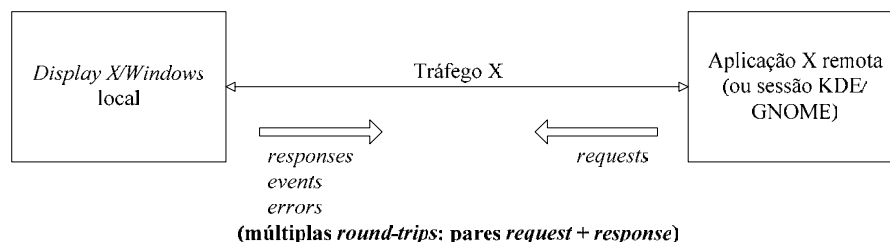



Figura 4.21 – *X/Windows*

O *X/Windows* apresenta também problemas de segurança, dado que não possui mecanismos de encriptação e protecção de tráfego. Ainda que seja possível acrescentar compressão e encriptação a uma conexão X11 remota, usando um túnel SSH seguro entre o cliente e o servidor (por exemplo com `ssh -C -X myusername@remoteserver "<aplicação>"`), este processo não opera de forma transparente.

Existe ainda outro problema para o qual este método não oferece solução: uma sessão remota X interactiva pode melhorar a fluidez e latência de resposta se se aumentar a largura de banda disponível entre o cliente e o servidor. No entanto, existe um ponto a partir do qual este acréscimo de capacidade deixa de beneficiar a velocidade da conexão, devido ao excesso de *round-trips* entre o cliente e o servidor.

Não é só o X11 que possui pontos fracos. Ainda que os protocolos RDP e VNC recorram a mecanismos mais eficientes em termos de uso de largura de banda (tais como algoritmos adaptivos de compressão diferencial), também carecem de mecanismos de protecção de tráfego.

Uma solução para estes problemas surgiu recentemente, pela mão da empresa *NoMachine* [Nomachine], que desenvolveu, com base em componentes *open source,* uma solução de acesso uniforme designada *NoMachine NX* (Figura 4.22). Agrupando diversos componentes herdados de projectos como o *cygwin* [Cygwin]*, rdesktop* [Rdesktop]*, openssh* [OpenSSH]*, openssl* [OpenSSL] e *VNC* [VNC], a *NoMachine* desenvolveu um protocolo que pode ser descrito sucintamente como uma conexão X11 que opera dentro de um tunel SSH, com uso de compressão e capaz de operar de forma eficiente mesmo sobre ligações com débito reduzido.

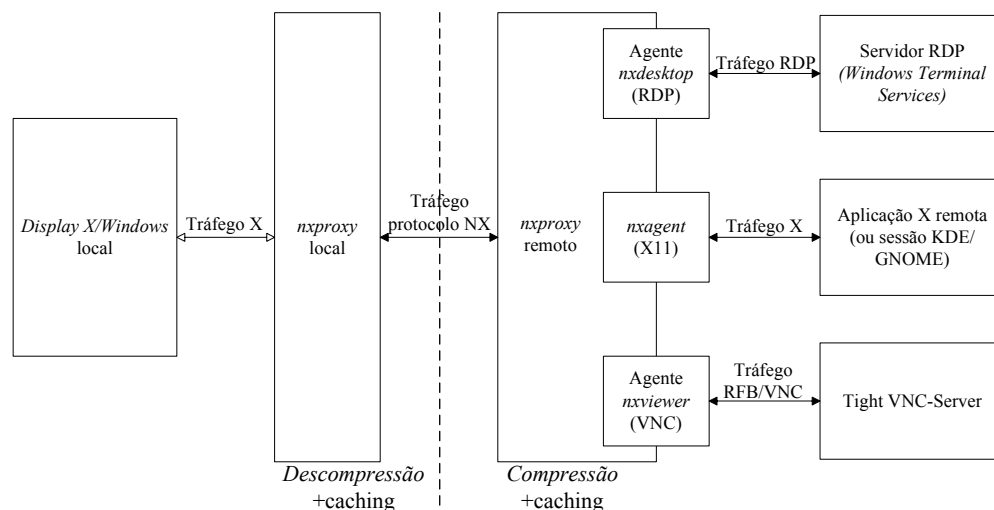



Figura 4.22 – Funcionamento do *NoMachine NX*

O protocolo NX permite reduzir de forma considerável o número de *round-trips* entre o cliente e o servidor, obtendo consideráveis ganhos de velocidade. Para isso recorre a dois mecanismos:

- **Compressão**. A *Nomachine* desenvolveu um algoritmo de compressão mais eficiente que a compressão ZLIB genérica (alegadamente dez vezes mais eficiente) e que utiliza apenas um décimo dos ciclos de CPU.
- ***Caching*** a fim de minimizar o número de transferências repetidas dos mesmos dados, ao mesmo tempo que restringe a transferência de dados similares a uma transacção diferencial (numa actualização de uma janela, por exemplo, apenas circulam os dados referentes às áreas que sofreram modificações). Isto ajuda a reduzir de forma significativa o número de *round-trips*.

Estes métodos permitem obter ganhos que podem ser chegar às 70x (conexão X11 remota com compressão máxima) em conexões com reduzida largura de banda disponível. Para os restantes protocolos, o *nxproxy* remoto está integrado com um conjunto de agentes que faz a translacção das conexões RDP e VNC para X11, para posterior tratamento. Isto permite obter, para esses protocolos, benefícios em termos de eficiência que podem chegar às 10x em termos de tráfego.

Devido a estas vantagens do protocolo NX optou-se por incluir na plataforma IC^3 um cliente *Nomachine NX* completo (nomeadamente o *FreeNX*). Este cliente pode estar pré-configurado para exportar sessões inteiras de *display* remoto ou apenas aplicações isoladas, de forma a garantir o acesso seguro e eficiente aos três protocolos suportados – X, VNC e RDP – mesmo sobre ligações de baixo débito. A Figura 4.23 ilustra o modo como a arquitectura NX se pode integrar na organização.

Refira-se ainda que não haveria perda de funcionalidade caso se optasse por incorporar os clientes tradicionais dos protocolos VNC e RDP (o X11 já é nativamente suportado) na imagem de sistema da plataforma IC^3, para operar nos moldes tradicionais. Os ganhos derivados do uso do NX são essencialmente ao nível de desempenho e eficiência.

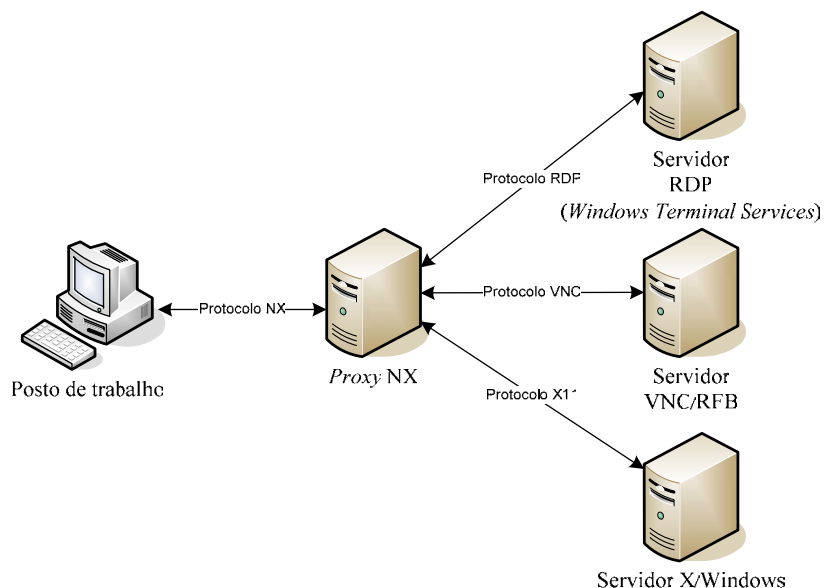



Figura 4.23 – Integração do Protocolo NX

4.5.6 Suporte VoIP e H.323

O suporte VoIP (protocolo SIP – *Session Initiation Protocol* [RFC3261]) e Videoconferência (H.323) são contemplados na arquitectura IC^3 como componentes essenciais para concretizar a consolidação num só dispositivo dos SoIP (*Services over IP*) e dos meios computacionais.

O suporte de videoconferência é implementado de forma simples, numa perspectiva de operação *peer to peer* (a arquitectura não contemplou o recurso a uma *gateway* H.323, se bem que a adopção desta seja uma mera questão de configuração), podendo ser integrado a nível dos clientes com o serviço de directoria LDAP para sincronização de *address book*. A aplicação utilizada para este efeito é o *gnomemeeting* [Gnomemeeting], com maturidade e interoperabilidade – decorrente da adopção do H.323 – adequadas aos propósitos em vista.

Se a adopção de um *gatekeeper/gateway* H.323 for considerada necessária, sugere-se o o uso do *Gnu Gatekeeper* [Gatekeeper], que suporta o acesso a servidores LDAP – para resolução de *alias* e autenticação – e o registo dinâmico de *alias* dos utilizadores de forma a poder lidar com uma estrutura de clientes móvel sem endereços IP fixos.

O suporte VoIP (SIP) é integrado por intermédio da aplicação *kphone* [Kphone] e é implementado tendo em vista o recurso a um SoftPBX *Open-Source* denominado *Asterisk* [Asterisk]. A integração com o serviço de directoria LDAP é feita a no servidor SoftPBX e não nos clientes, ainda que estes possam manter uma lista de contactos própria a nível local. O *roaming* de utilizadores (conforme previsto na Secção 3.4.1) é conseguido graças a um *dialplan* (que permite parametrizar extensões e serviços de telefonia a nível do PBX) especificamente configurado a nível do SoftPBX de modo a contemplar a existência de extensões telefónicas VoIP sem um endereço IP/Posto de trabalho fixo associado (Figura 4.24).

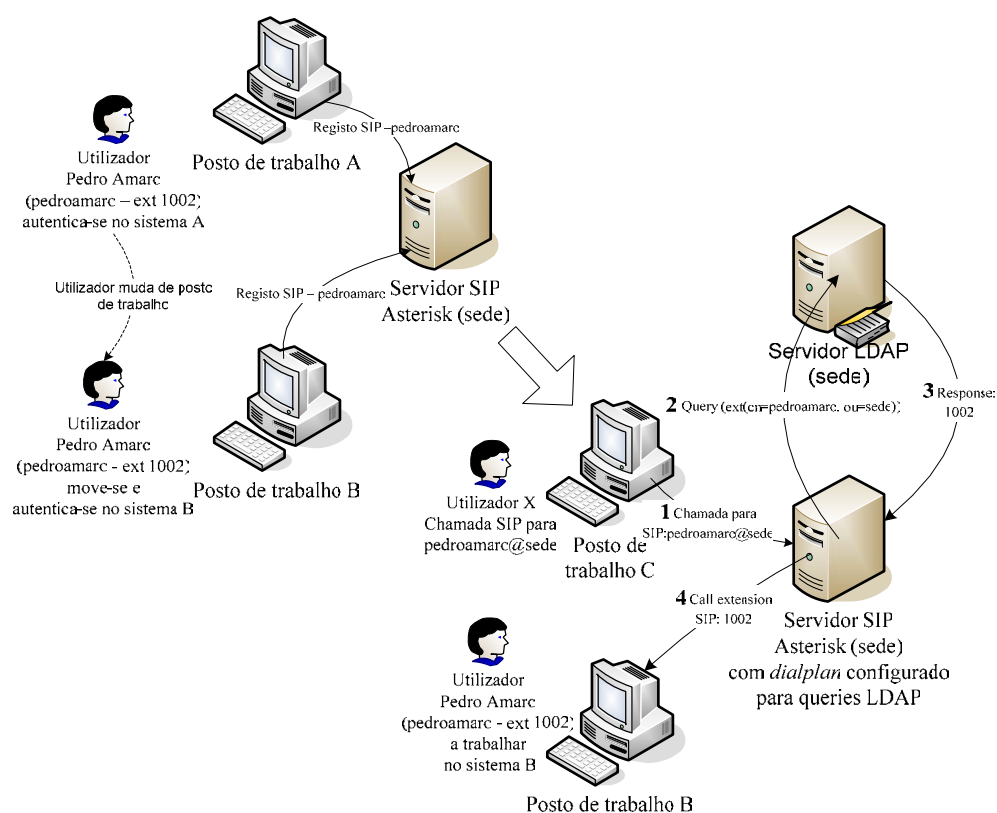



Figura 4.24 – *Roaming* de Utilizadores para Serviços VoIP

Para esse efeito recorreu-se a uma *macro*, no *dialplan* do SoftPBX, que instancia dinamicamente o identificador por meio de uma regra que permite efectuar interrogações ao servidor LDAP (de forma independente do esquema LDAP adoptado, pois todo o filtro de busca é parametrizável) para fazer a resolução do nome do utilizador numa extensão. O inverso é também possível, por exemplo para implementar a função de CALLERID. Deste modo, quando o utilizador se move de um posto de trabalho para outro, a sua extensão telefónica acompanha-o.

O *Asterisk* suporta também a funcionalidade de *gateway* H.323 para efeitos de serviço VoIP, embora o recurso a esta possibilidade não seja considerado na arquitectura desenvolvida.

A arquitectura proposta tem a vantagem de ser escalável (Figura 4.25), permitindo expandir a solução de telefonia VoIP para várias filiais de uma empresa, cada uma com o seu SoftPBX conectado ao da sede por um *trunk IAX* (*Intra Asterisk eXchange*), sobre uma ligação TCP/IP dimensionada para o efeito (podendo mesmo recorrer-se a VPNs, se tal for conveniente).

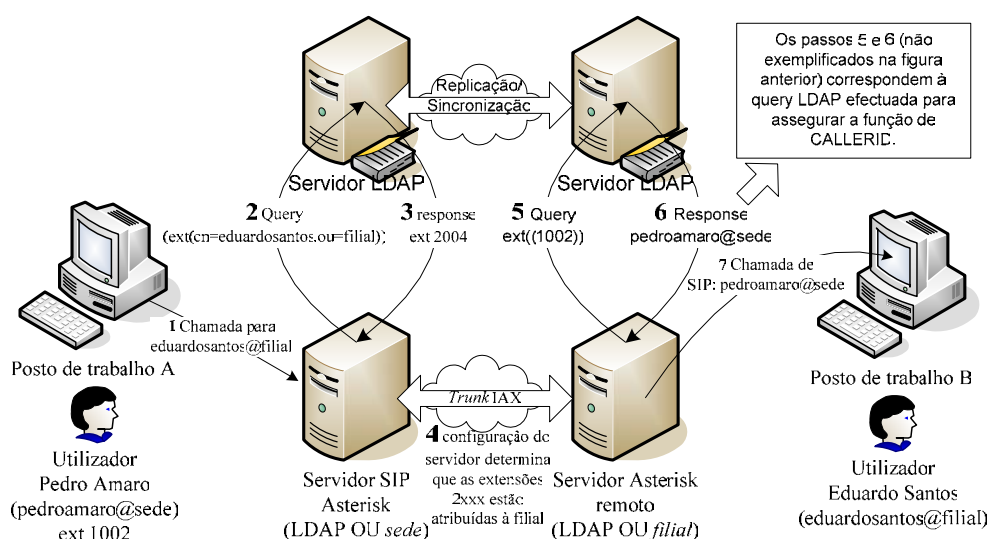



Figura 4.25 – Escalabilidade da Arquitectura VoIP Proposta

Note-se no entanto que persiste ainda uma fragilidade nesta solução, relacionada com a autenticação dos utilizadores aquando do seu registo. Cada utilizador autentica-se no SoftPBX com recurso a um par *username+password*. Contudo, como a comunicação com o servidor LDAP é feita em *cleartext*, não será recomendável armazenar as *passwords* dos utilizadores no serviço de directoria, tendo estas de ser mantidas no ficheiro local de configuração do SoftPBX.

Uma solução possível mas pouco elegante passa por, a intervalos regulares, ligar ao servidor LDAP por intermédio de um tunel SSH ou SSL, extraindo as *passwords* em *cleartext* ou os *hashs MD5* destas (que também podem ser utilizados para autenticação pelo *asterisk*), com um *script* que se encarregaria de actualizar o ficheiro de configuração do PBX.

Outra solução, mais elegante, consiste no recurso ao suporte RADIUS do *Asterisk* para autenticar os utilizadores de forma segura (utilizando os servidores RADIUS como intermediários) com base na informação existente no servidor LDAP. Esta última solução não foi testada no âmbito do IC³, embora esteja documentada [Sultan 2005].

O Anexo B discute de forma mais detalhada estes e outros aspectos relacionados com o suporte VoIP na plataforma IC³.

4.5.7 Aplicações

A escolha das aplicações a incorporar na imagem de sistema IC^3 obedece a um conjunto de critérios bem-definidos e intimamente relacionados com o propósito do protótipo. Ainda que o recurso a compressão dinâmica permita dispor de 1,2 GByte de espaço utilizável, a *payload* de aplicações e utilitários a incorporar na imagem de sistema foi alvo de algum refinamento, de modo a optimizar o espaço utilizado. Do conjunto de aplicações escolhidas destacam-se:

- ***Web Browser.*** *Mozilla-firefox* com plugins *Flash e Java* pré-configurados. A escolha deste *browser* deve-se ao seu desempenho e estabilidade, sendo um dos *browsers* mais utilizados em plataformas *não-windows*. É suportado também o *konqueror* (motor KHTML), incorporado no ambiente KDE que, apesar de alguma falta de maturidade, se constitui como uma alternativa leve e rápida ao *firefox*.

- **Aplicações de Produtividade.** *OpenOffice 1.3*, que suporta os formatos de documentos mais correntes e exportação para PDF. Inclui aplicações para processamento de texto, folha de cálculo, apresentações e desenho vectorial (ilustrações), estando prevista para a versão 2.0 (em fase Beta) uma aplicação de base de dados análoga ao *Microsoft Access* (além de uma melhoria assinalável na velocidade da execução e resposta dos componentes da *office suite*). Nas versões *light* o *OpenOffice* pode ser substituido pelo *Koffice* ou *abiword* com *xcalc*, ainda que com alguma perda de funcionalidade.

- **Clientes de Correio Electrónico.** *Mozilla-thunderbird* e *Evolution*. Ambos os clientes suportam acesso a *mailboxes* POP ou IMAP e *plugins* para interoperar com soluções de *groupware*. O *mozilla-thunderbird* é suportado devido à sua estabilidade e simplicidade, estando configurado de origem com o *plugin Enigmail* para suporte de encriptação e assinaturas OpenPGP.

- **PIM (*Personal Information Management*).** *Evolution*. Além de se constituir como um bom cliente de correio electrónico, o *Evolution* oferece funcionalidades PIM com recurso a um *interface* acessível e intuitivo.

- ***Media Players.*** *xine* e *kaffeine* (suporte MP3, DivX, DVD configurado e activado).

- **Gravação de CDs e DVDs.** Aplicação *cdrecord* com *frontend* K3B. O K3B fornece um *interface* intuitivo e familiar para o subsistema de gravação de CDs/DVDs.

- **Aplicação de Videoconferência.** *Gnomemeeting.* Esta escolha deve-se ao facto de o *Gnomemeeting* ser considerado o cliente de videoconferência H.323 mais maduro, estável e interoperável na plataforma *Linux.*

- **Cliente VoIP SIP:** *Kphone* e/ou *SJPhone.* Apesar da versatilidade do *SJPhone*, a preferência é dada do *Kphone* por dois motivos: boa integração com o ambiente KDE (minimiza para ícone na *toolbar* e é discreto) e licenciamento gratuito. O *SJPhone* é suportado para contemplar os casos em que os benefícios de flexbilidade acrescidos pela solução compensem os custos de licenciamento.

- **Desenho Vectorial.** *DIA*, uma ferramenta de desenho vectorial bastante completa cujo único problema – ausência de bibliotecas de componentes – é compensado com um tamanho reduzido, estabilidade e rapidez de funcionamento.

- **Acesso a Sistemas Remotos** (cfr. Secção 4.5.5). Cliente *FreeNX* ou, em alternativa, os clientes *rdesktop* (para suporte RDP) e *vncclient* (para suporte VNC). Aos protocolos tradicionais de *display* gráfico remoto acresce ainda o suporte para o protocolo *Citrix Metaframe* (cliente incorporado em opção) e para cliente de terminal orientado a texto com as emulações mais correntes (incluindo 3270).

- ***Software* de Edição de Imagem.** Para este efeito é incluído o *GIMP*, uma ferramenta de edição e manipulação de imagem cuja amplitude de funcionalidades satisfaz um leque de utilizadores bastante abrangente.
- ***Software* de *Instant Messaging*.** O sistema inclui uma ferramenta de *instant messaging* chamada GAIM, que suporta os protocolos mais difundidos deste género, com o ICQ, MSN, AIM, *Jabber*, *Yahoo!* e *Groupwise* (entre outros).

Estas são apenas algumas das aplicações incorporadas na imagem de sistema, às quais ainda se poderiam acrescentar, entre outras, sistemas de *Desktop Publishing* como o *Scribus* [Scribus] e ferramentas de gestão e monitorização de rede.

4.6 Síntese da plataforma

Até ao momento, a abordagem adoptada neste capítulo tratou o projecto IC^3 numa perspectiva *bottom-up*, partindo do nível *hardware* até chegar aos serviços e aplicações, passando pelo sistema e ambiente de operação. Devido ao nível de detalhe técnico necessário para tratar cada um dos aspectos envolvidos considerou-se que esta aproximação seria a mais adequada, visto permitir decompor o tratamento da implementação do protótipo IC3 numa base modular. Nesta secção procura-se apresentar uma visão de conjunto que agregue de forma coerente as peças/modulos discutidos previamente, criando deste modo uma perspectiva complementar à centrada nos blocos funcionais (e que corresponde às secções anteriores deste Capítulo), individualmente.

4.6.1 A arquitectura do sistema IC^3 vista numa perspectiva global

A arquitectura proposta e implementada no protótipo IC^3 pode ser sintetizada com a ajuda de um diagrama a três níveis funcionais: *hardware, software* de sistema e nível aplicacional/serviços, conforme mostra a figura 4.26.

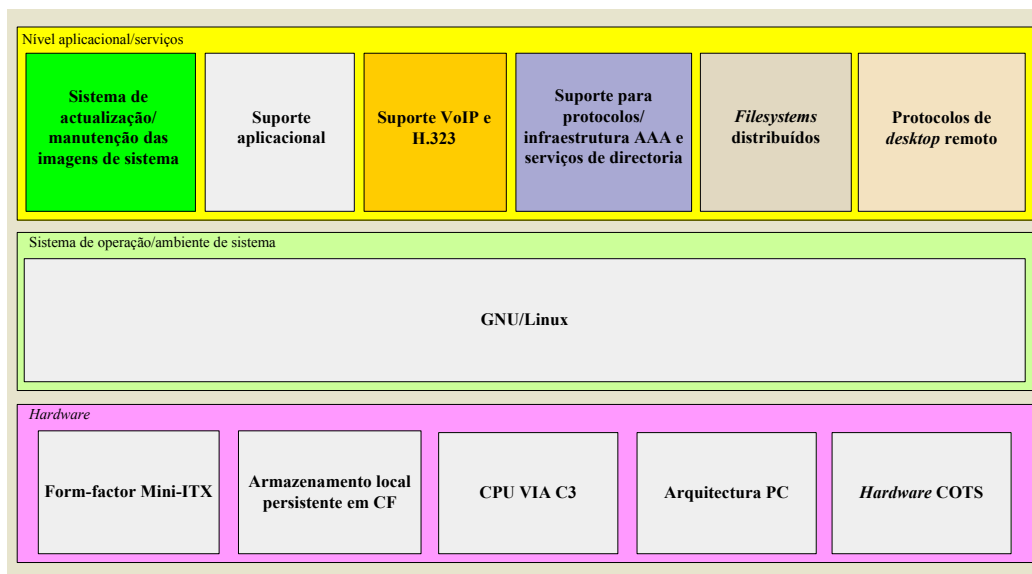



Figura 4.26 – O arquitectura do sistema IC^3

Cada nível funcional encontra-se decomposto em módulos individuais. Evitando uma descrição exaustiva, optou-se por dar relevância aos componentes que constituem, de algum modo, inovação ou possuem especial importância no contexto específico do projecto.

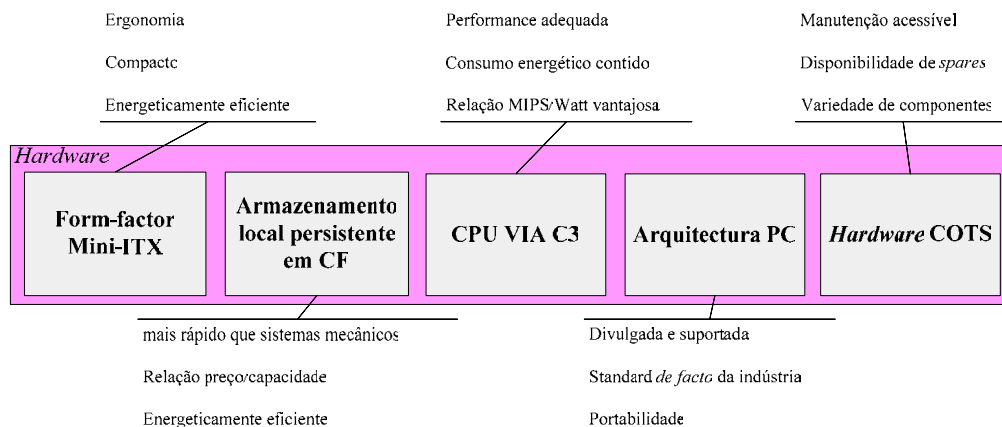



Figura 4.27 – A arquitectura do sistema IC^3: nível do *hardware*

A nível do *hardware* (Figura 4.27) deu-se especial importância a um conjunto de aspectos, nomeadamente:

- ***Form-factor.*** Na secção 4.2.2 discutiram-se as razões que levaram à adopção do formato Mini-ITX, tendo esta opção sido determinante para a construção de um sistema compacto, silencioso e energeticamente eficiente.
- **Armazenamento local baseado em CF.** A adopção de armazenamento de massa local (para a imagem de sistema – cfr. Secção 4.2.3) baseada em tecnologia *Compact Flash* permitiu opter ganhos em termos de desempenho quando comparado com os sistemas mecânicos ópticos tradicionais (e mesmo contra os magnéticos há que lembrar que o CF possui maior consistência e uniformidade em termos de *performance*), para além de uma fiabilidade acrescida e baixo consumo energético. O limite de escritas da tecnologia flash não é um factor relevante neste projecto visto o sistema estar concebido para operar com base num volume *read-only*.
- **CPU VIA C3.** Este processador é uma implementação energeticamente eficiente de uma arquitectura compatível x86 cujo binómio performance-energia consumida (cfr. Secção 4.2.1) o coloca como uma escolha perfeitamente adequada para os propósitos deste projecto. Indirectamente, esta escolha acaba por beneficiar o aspecto da ergonomia, devido ao facto do processador possuir um reduzido envelope térmico e não requerer uma grande capacidade de dissipação activa.
- **Arquitectura compatível PC.** A opção por esta arquitectura (cfr. Secção 4.2.4) permite beneficiar quer de um grande leque de *hardware* disponível, quer de um amplo conjunto de aplicações especificamente produzidas para esta plataforma.
- ***Hardware* COTS.** A adopção de *hardware* corrente permite dispor de uma variedade de opções em termos de custo e *performances* que vem beneficiar o custo de manutenção e actualização do sistema (e indirectamente, o seu TCO).

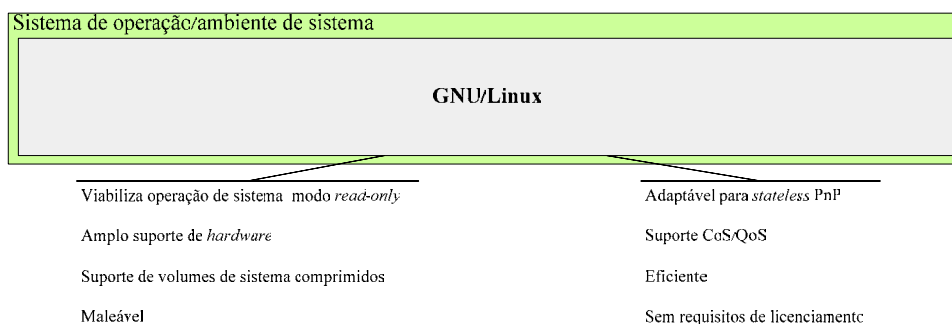



Figura 4.28 – A arquitectura do sistema IC^3: sistema de operação/ambiente de sistema

A adopção do sistema operativo GNU/Linux para o ambiente de sistema (Figura 4.28) aligeirou o ciclo de desenvolvimento do protótipo e possibilitou um conjunto de benefícios de grande relevância:

- **Maleabilidade.** Foi a grande maleabilidade do sistema Linux que tornou possível a criação de um ambiente capaz de operar a partir de um volume comprimido em tempo real, em modo *read-only* com a possibilidade de operar em modo *plug-and-play stateless*, recorrendo apenas a ferramentas e meios livremente disponíveis (cfr. Secção 4.4).

- **Suporte de hardware.** O *kernel* Linux suporta um vasto leque de componentes de hardware, o que permitiu que fosse possível optar pelo recurso a componentes convencionais (e, por consequência, disponíveis em volume e variedade).

- **Licenciamento.** Sendo livre de licenciamento, o sistema Linux permitiu enveredar pelo denvolvimento da plataforma sem a necessidade de considerar custos de licenciamento e/ou acordos NDA.

- **Suporte QoS/CoS.** Graças ao suporte CoS/QoS nativo do sistema, tornou-se possível desenvolver mecanismos de prioritização de tráfego (cfr. Secções anexas B2.3, B2.4 e B2.5) tendo em vista a adequação do desempenho do sistema IC3 no contexto da convergência de comunicações de dados, voz e imagem sobre uma rede IP.

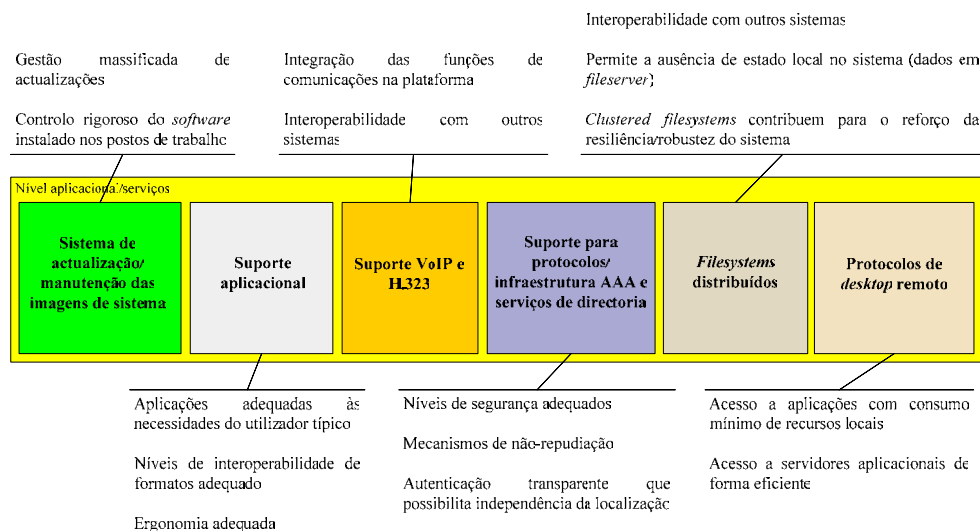



Figura 4.29 – A arquitectura do sistema IC³: nível aplicacional/serviços

Na camada aplicacional/serviços (Figura 4.29) temos um conjunto de módulos funcionais, dos quais se destacam:

- **Actualização distribuída de imagens de sistema.** Este sistema, suportado por um conjunto de procedimentos claros e bem definidos torna a difusão das actualizações das imagens de sistema pelos postos de trabalho num processo seguro e eficiente (cfr. Secções 4.5.1 e 4.5.2).

- **Suporte/bundle aplicacional.** O conjunto de aplicações seleccionadas para a imagem de sistema do protótipo cobre as necessidades típicas do utilizador convencional, Secção podendo contudo ser personalizado à medida dos perfis de utilizadores existentes (cfr. Secção 4.5.7).

- **Suporte VoIP e H.323.** Aliado aos mecanismos de directoria, estas características permitem a convergência da funcionalidade de comunicação (telefonia e/ou vídeoconferência) com o meio computacional em si, suportando o roaming dos utilizadores (cfr. Secção 4.5.6.).

- **Suporte para infraestrutura AAA e serviços de directoria.** Aliado a uma infraestrutura AAA adequada, o sistema cumpre os requisitos de segurança, não-repudiação e transparência da localização definidos para o projecto (cfr. Secção 4.5.3).
- ***Filesystems* distribuídos.** O suporte para uma ampla variedade de sistemas de ficheiros distribuídos permite a concretização de dois aspectos vitais: a interoperabilidade e troca de informação com um vasto leque de plataformas e a ausência de estado local, visto os dados dos utilizadores estarem acessíveis via rede. Simultaneamente, a adopção do suporte para *filesystems* com capacidades de redundância e transparência de localização permitem reforçar a resiliência e fiabilidade da arquitectura (cfr. Secção 4.5.4).
- **Protocolos de desktop remoto.** Da mesma forma que o suporte de *filesystems* distribuídos permite a interoperabilidade em termos do acesso a dados presentes noutras plataformas, o suporte para protocolos de *desktop* remoto (cfr. Secção 4.5.5) permite o acesso a aplicações existentes nessas mesmas plataformas de forma eficiente em termos do uso de largura de banda e uniforme (caso se recorra ao protocolo NX).

Em jeito de resumo, não será incorrecto afirmar que a perspectiva sintética e simplificada da plataforma IC^3 que é apresentada nesta secção mostra uma arquitectura que concretiza os requisitos funcionais enunciados nas Secções 3.3 e 4.1, recorrendo a tecnologias, normas e ferramentas acessíveis, existentes e divulgadas, que apenas careciam da devida integração para formar um todo coeso.

4.7 Validação do conceito

No sentido de validar o conceito subjacente ao projecto IC^3 procedeu-se à construção de três protótipos (um móvel e dois do tipo *desktop*). De seguida, procedeu-se à integração de um protótipo de cada tipo num ambiente real (a rede da Associação de Informática da Região Centro) durante um período limitado de tempo, onde se submeteram os protótipos a uso e apreciação por parte de alguns dos utilizadores. No respeitante à integração na rede e serviços telemáticos já existentes, o caso estudado constitui a base para a informação apresentada na Figura 4.30 (e posteriormente, na secção 5.3 quando se tratar a questão da mobilidade).

Numa primeira linha surge o suporte para os serviços AAA (cfr. Secção 4.5.3), concretizado com o apoio de um ou vários servidores de directoria LDAP que disponibilizam – além de informação sobre postos de trabalho, periféricos e utilizadores – estruturas necessárias à operação normal da arquitectura de serviços como os *automount maps* (conforme descrito na Secção 4.5.4) e as extensões SIP (cfr. Secção 4.5.6). Estes servidores servem adicionalmente de *backend* de autenticação para o serviço RADIUS (cfr. Secção 4.5.3), necessário para o processo de autorização 802.1X e *accounting*. Este nível assegura a uniformidade das contas dos utilizadores e dos processos de autenticação ao longo da estrutura gerida, sendo a base de sustentação dos mecanismos de *roaming*.

Na segunda linha de servidores surge a infrastrutura de apoio[5] onde se alicerça a operação dos postos de trabalho, e que fornece essencialmente três categorias de serviço:

- SoIP (cfr. Secção 4.5.6);
- *filesystem* distribuído (previamente discutido nas Secções 4.5.2, 4.5.4, 5.2.1 e 5.2.2);
- e *display* remoto (cfr. Secção 4.5.5).

[5] Para efeitos de simplificação da figura omitem-se os servidores de DNS, DHCP, HTTP, *Proxy* HTTP/SOCKS e afins, que não estão directamente relacionados com as especificidades da arquitectura e que pouco ou nenhum valor acrescentam à descrição aqui efectuada.

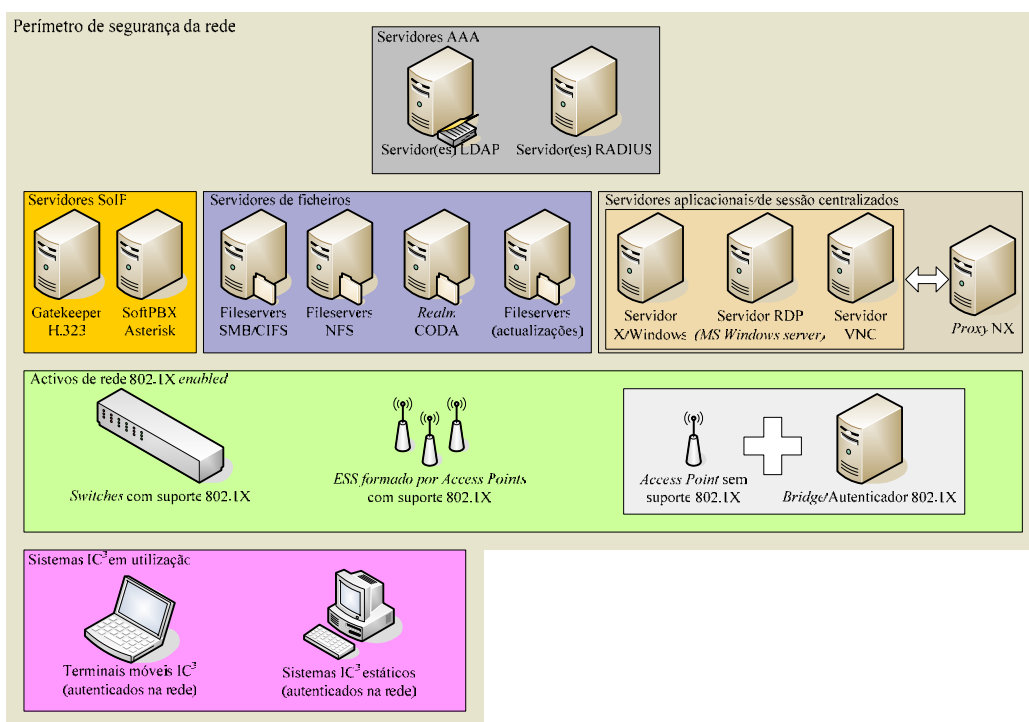



Figura 4.30 – Infraestrutura da Plataforma IC³
(inclui o suporte de mobilidade a tratar na Secção 5.3)

Para as duas primeiras categorias a integração com LDAP permite uniformizar o processo de gestão das contas de utilizador – exceptuando o caso já mencionado dos servidores Coda – e satisfazer as necessidades de segurança no respeitante à não repudiação das suas actividades. É também esta integração que permite que um utilizador tenha acesso ao seu ambiente de trabalho de forma automática e transparente, onde quer que seja que se autentique, pois a sua área de trabalho será montada automaticamente, graças ao *automount map* do utilizador[6].

O processo que permite a redirecção da extensão telefónica do utilizador para o posto onde este se autentica é também possível, graças à integração entre o SoftPBX e o serviço de directoria LDAP, que contém as associações entre um dado utilizador e a extensão do PBX correspondente.

Para a categoria de servidores de *display* remoto – que fornecem acesso a aplicações exportadas e a sessões completas – são contempladas duas hipóteses (cfr. Secção 4.5.5):

- acesso directo aos serviços com recurso a cliente nativo incorporado nos postos de trabalho;
- ou acesso por intermédio do protocolo NX, através do *Proxy* NX que fará a mediação entre os clientes e os servidores.

Por meio do suporte a estes protocolos torna-se possível trabalhar com aplicações que apenas executam noutras plataformas – como será o caso das aplicações Windows – ou cuja

[6] A este nível, note-se ainda que para os serviços de fileserver SMB é possível utilizar servidores UNIX com suporte SMB (utilizando o serviço Samba) ou sistemas Windows. Contudo, neste último caso existirá trabalho adicional a desenvolver, não contemplado nesta dissertação, para assegurar a uniformidade do processo de autenticação com o resto da infraestrutura. No respeitante ao protocolo SMB existe uma limitação adicional, relacionada com a inexistência de suporte para tipos de ficheiros específicos do UNIX (por exemplo *named pipes* e *links* simbólicos), tornando-o desaconselhado para armazenar informação dos perfis dos utilizadores, ainda que possa ser utilizado sem problemas significativos para armazenar os seus dados.

utilização se encontra reservada apenas a este tipo de acesso por motivos de segurança, limitações de largura de banda ou peso computacional.

Na terceira linha da infraestrutura encontram-se os activos de rede, que para além do fornecimento da conectividade de rede desempenham o papel de *gatekeepers* 802.1X, controlando as autorizações de acesso ao interior do perímetro seguro da rede da organização em articulação com os servidores RADIUS do primeiro nível (cfr. Secção 4.5.3). Prevê-se a este nível o suporte para *Access Points* sem capacidade de autenticação 802.1X, recorrendo para o efeito à interposição entre estes equipamentos e os activos de rede de um servidor *Linux* com a *package hostapd*.

Finalmente, no último nível surgem os postos de trabalho IC^3 (móveis e estáticos/*wired*) cuja operação é assegurada pelos três primeiros níveis de serviço.

4.7.1 Notas gerais sobre o processo de integração do conceito IC^3

Deste processo são de salientar as seguintes observações:

- Apesar da predominância do paradigma WIMP (*Windows, Icons, Mouse and Pointer*) e da ergonomia dos ambientes gráficos em Linux ter vindo a melhorar de forma significativa, muitos utilizadores habituados à plataforma *windows* padecem de dificuldades (em grau variável, convenha-se em abono da verdade) para familiarizar-se com a nova plataforma.
- A integração do conceito IC3 numa infraestrutura já existente exige um estudo cuidado no sentido de aferir o modo como este se pode integrar nos serviços já existentes e de que forma pode servir as necessidades específicas dos perfis dos utilizadores existentes na organização.
- Nem sempre será viável implementar todos os serviços previstos na arquitectura de referência, seja por carecerem de importância no contexto específico da organização, ou pelos benefícios não justificarem o custo de implementação e manutenção.
- Os perfis de utilizadores devem ser identificados e classificados antes da integração de um parque de sistemas IC3, de modo a aferir e identificar os melhores candidatos para a migração para os novos postos de trabalho. No estágio actual, a solução IC3 não se adapta facilmente a *power-users* com hábitos de trabalho enraízados.
- Quando o acesso a aplicações em outras plataformas for uma prioridade o responsável pela gestão dos sistemas deverá ter sempre em conta a carga acrescida colocada nos servidores aplicacionais como consequência da adição de novos sistemas cliente (e.g., Windows 2003 com *terminal services*) e o licenciamento por sessão concorrente, quando pertinentes.
- Os utilizadores não deverão ser postos à parte no respeitante ao processo de migração – o *feedback* destes deve ser valorizado e incentivado visto o sucesso da integração do conceito IC3 depender em larga medida do cuidado posto na adaptação da solução ao caso específico de cada organização.

5. Suporte para Mobilidade

O Capítulo anterior foi dedicado à apresentação e discussão da plataforma IC^3, abrangendo o *design* e arquitectura do protótipo, a integração dos diversos componentes e a operação do sistema.

Neste Capítulo é aprofundado um dos aspectos mais importantes da plataforma: o suporte para mobilidade. Neste contexto, serão discutidas as especificidades do modo de operação da plataforma em ambientes móveis, com especial ênfase para suporte de *disconnected computing* e operação em modo *offline*.

Em paralelo, neste Capítulo será também apresentado um segundo protótipo de *hardware* mais adequado para ambientes móveis – baseado na modificação de um *notebook* – e serão destacadas as diferenças entre este protótipo e a versão *desktop* da plataforma.

5.1 Protótipo Móvel: Soluções de *Hardware*

O primeiro desafio que se colocou na construção de um protótipo móvel foi a adaptação de um *notebook* tradicional, de modo a que pudesse operar de modo semelhante ao protótipo *desktop*. Para esse fim usou-se um adaptador CF-IDE de 2,5" passível de instalação no local reservado para o disco rígido, sem necessidade de modificações adicionais (Figura 5.1).

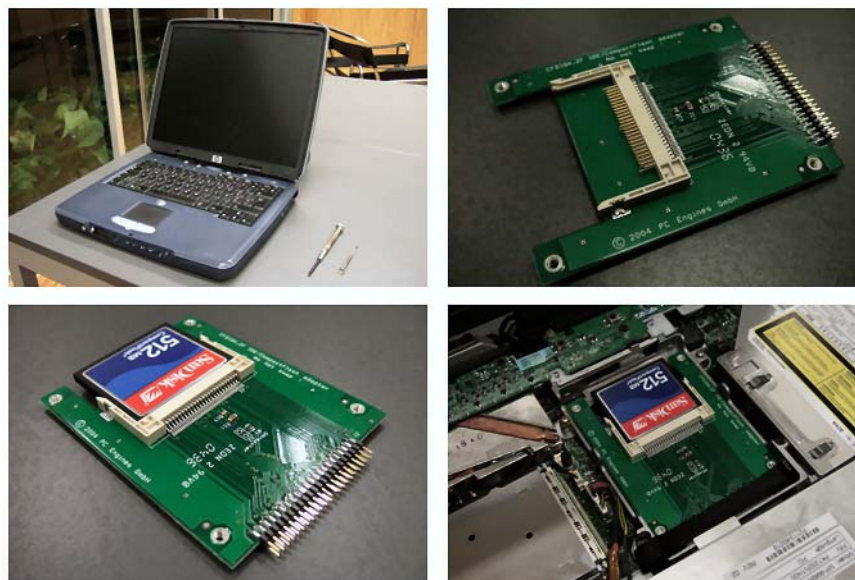



Figura 5.1 – Substituição do Disco Rígido por Cartão CF

Este adaptador opera de modo idêntico ao do modelo descrito na Secção 4.2.3, permitindo que o cartão CF seja reconhecido e acedido pelo sistema como se fosse simples disco IDE/ATA. Esta adaptação acarreta benefícios para o próprio *notebook*, em termos de robustez e economia de energia (*cfr. caixa na página seguinte*). As modificações contemplaram ainda o acrescento de um adaptador CF-PC Card (Figura 5.2), de modo a dispor de um segundo cartão amovível no sistema. Este cartão – que será designado por *synchrocard* – servirá para manutenção de informação vital do sistema e/ou do utilizador, quando o *notebook* estiver em modo *disconnected*.

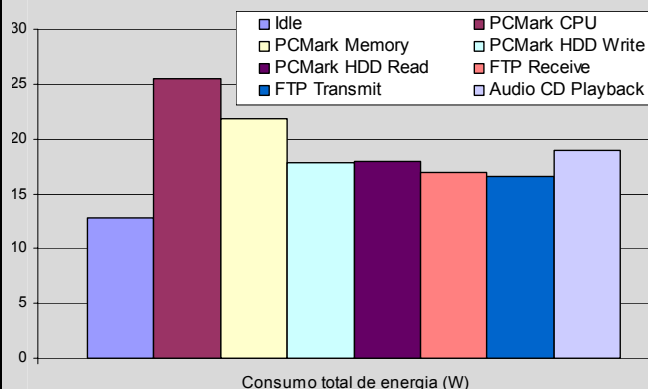
Figura 5.2 – Instalação de Adaptador CF-PC Card

Consumo de Energia do Disco Rígido num *Notebook*

De acordo com um estudo efectuado com o objectivo de determinar o consumo de energia por componente, num notebook moderno [Mahesri 2004] – um IBM *ThinkPad* R40 – obtiveram-se os seguintes resultados:

Nível de Brilho do LCD	Consumo de Energia
1	0,5W
4	1,25W
8	3,5W

Estado do Disco Rígido	Consumo de Energia
Idle	0,575W
Standby	0,173W
Read	2,78W
Write	2,19W
Copy	2,29W

Estado da Unidade Óptica	Consumo de Energia
Spin Up	3,34W
Steady Spin	2,78W
Read	5,31W

Frequência da CPU	*Idle*	*Max*
600 MHz	14,31W	16,54W
800 MHz	15,69W	20,98W
1000 MHz	15,88W	22,71W
1200 MHz	16,47W	25,71W
1300 MHz	16,9W	27,45W

Modo do adaptador *Wireless*	Consumo de Energia
Powersaver	0,14W
Base (Idle)	1,0W
Transmit	3,12W (4,2Mb/s)
Receive	2,55W (2,9Mb/s)

Conforme o padrão de uso, o disco rígido é responsável por uma percentagem do consumo total de energia do sistema que pode ir de 3 a 15%. Outros estudos similares [Lorch 1995][Douglis 1993] confirmam estas observações, mesmo para sistemas e equipamento de gerações anteriores (*notebooks* baseados em CPUs i386SL, 486SL e 680x0).

A substituição do disco rígido tradicional de 2,5", responsável por consumos de energia da ordem dos 2W (em operação) por um cartão CF, cujo consumo ronda os 0,5W (equivalente ao consumo do disco rígido em modo *idle* ou *standby*) acarreta um ganho bruto aproximado de 1,5W.

Além das reduções no consumo de energia e consequente aumento de autonomia, o cartão CF aumenta a fiabilidade (menor número de componentes mecânicos, em geral mais permeáveis a falhas), reduz o *stress* do sistema (um cartão CF constitui, sob várias formas, um componente mais "amigável" no respeitante à sua relação com o resto do sistema, por motivos que vão da menor dissipação de calor à redução do *stress* mecânico) e melhora a ergonomia do sistema (redução de ruído e calor emitido).

O *synchrocard* constitui uma adição discreta que, uma vez instalada num *slot* PCCard se integra no chassis do *notebook* sem saliências visíveis. O cartão é detectado e utilizado pelo sistema como um disco IDE/ATA normal, utilizando o controlador IDE/ATA PCCard/PCMCIA embebido no cartão CF para *interface* (cfr. Secção 4.2.3).

Em futuras versões do protótipo será possível adicionar suporte para os leitores de *flash cards* integrados em alguns *notebooks* (Figura 5.3), permitindo o uso de cartões de formato MMC (*MultiMedia Card*), SD (*Secure Digital*) e MG/MS (*Magic Gate*/*Memory Stick* – podendo eventualmente tirar-se partido das funções de DRM existentes em alguns cartões desta família) para a função de *synchrocard*.

Outra alternativa a considerar, para situações em que se pretenda que o terminal móvel seja totalmente autónomo e não haja interesse no uso de um cartão amovível, será substituir o *synchrocard* por uma partição do cartão CF principal de sistema, que neste caso deverá ter uma capacidade superior aos presentes 512 MByte. Esta solução implica no entanto um maior desgaste do cartão principal de sistema, com a eventual necessidade de substituição ao fim de algum tempo.

Figura 5.3 – Leitores de Memória Flash Incorporados

5.2 Adaptação do Ambiente de Sistema IC^3 para Mobilidade

No respeitante às questões especificamente relacionadas como a mobilidade dos terminais, um dos grandes desafios que se colocou ao desenvolvimento da solução IC^3 foi manter um compromisso equilibrado entre a ausência de estado local persistente e a operação em modo *disconnected* – cujo suporte se afigura vital, dadas as características especificas no respeitante à fiabilidade da conectividade de rede *wireless* e a vocação dos sistemas móveis.

Independentemente das soluções adoptadas, tornou-se evidente que deveria existir um meio persistente de armazenamento dos dados dos utilizadores para operação *off-line* permanente ou transiente (ou seja, durante as falhas de conectividade de rede) sendo esta função desempenhada pelo já mencionado *synchrocard*. Este cartão é amovível, podendo ser removido e transportado pelo utilizador para outro terminal móvel IC^3, de modo a prosseguir com o seu trabalho – um compromisso aceitável entre a ausência de estado local e a operação em modo *disconnected*.

A operação em modo *disconnected* decorre de duas formas – explícita e implícita – conforme o grau de conhecimento e envolvimento que o utilizador tenha com a gestão da mobilidade do terminal.

5.2.1 Operação Explícita do *Disconnected Mode*

Este é o modo de operação mais simples, assumindo-se que é o utilizador a iniciar e tratar do processo de criação da *cache* móvel (no *synchrocard*) e da sua posterior sincronização com os dados existentes na *roaming home*.

A Figura 5.4 ilustra o funcionamento do modo *disconnected* em operação explícita. Após o processo de autenticação e associação de rede, o utilizador inicia o processo de criação da *cache* persistente no *synchrocard* (através da execução de um *script* criado para o efeito existente no ambiente de sistema), fazendo um *rsync* [Rsync] dos dados pertinentes da *roaming home* (perfil do utilizador, ficheiros contendo a configuração de componentes software/serviços e a *roaming briefcase*, análoga à pasta dos documentos dos sistemas *windows*).

De seguida, e se o processo for bem sucedido, o utilizador inicia a sua sessão de trabalho em meio persistente. No momento de ressincronizar os dados o utilizador desencadeará explicitamente o processo, por meio do mesmo *script*.

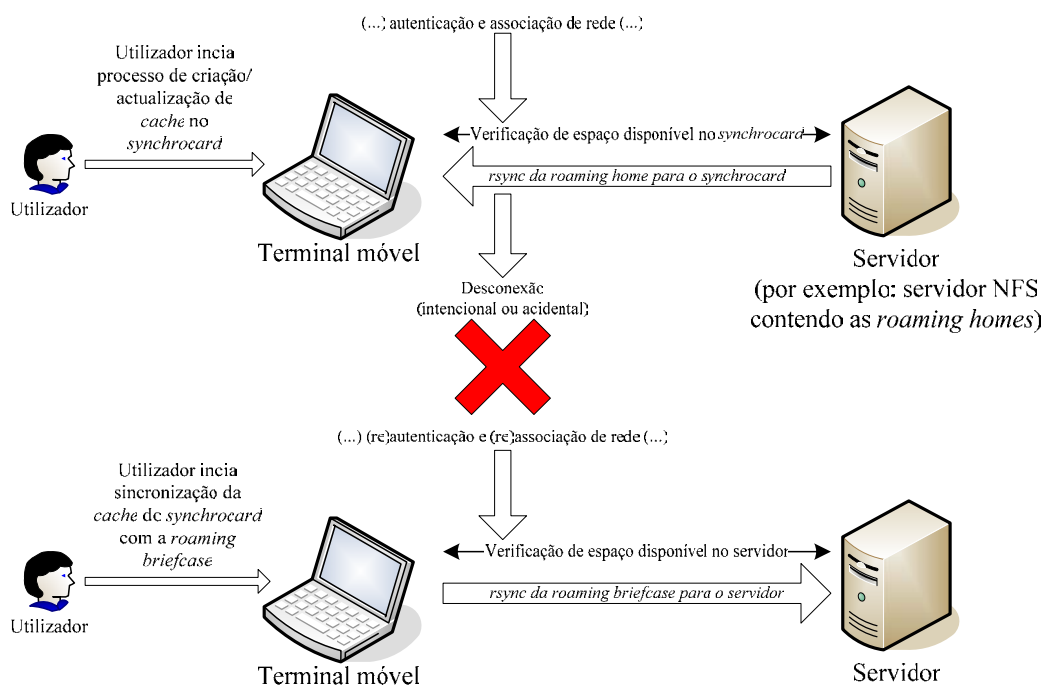



Figura 5.4 – Operação Explícita do *Disconnected Mode*

No respeitante à ressincronização dos dados, o *rsync* encarrega-se de conduzir o processo de forma eficiente, recorrendo a mecanismos internos de *pipelining* que permitem reduzir a latência das operações e, caso um dado ficheiro exista de ambos os lados, transmitindo apenas as diferenças entre as duas versões. O *rsync* é capaz de operar sobre *ssh, rsh* ou *sockets* ponto--a-ponto, fazendo uso de mecanismos de correcção de erros e sendo capaz de transferir uma árvore completa de directórios preservando permissões e posse dos ficheiros, *links*, dispositivos, e *timestamps*. Neste cenário, a resolução de potenciais conflitos – passíveis de surgir se o utilizador efectuar um acesso à *roaming briefcase* com modificação de dados, a partir de outro terminal, durante o período de desconexão) é deixada ao critério e responsabilidade do utilizador.

Aquando do arranque o terminal móvel IC³ efectua a detecção do *synchrocard*. Se este não estiver vazio e os dados nele existentes não pertencerem ao mesmo utilizador que está autenticado no terminal o seu uso é impedido, sendo o utilizador notificado da ocorrência (a

reinicialização de um *synchrocard* só deve ser autorizada pelo utilizador que o inicializou anteriormente ou pelo administrador de sistema).

Se o *synchrocard* estiver vazio ou preenchido com dados do utilizador que se autenticou no sistema, este poderá prosseguir com o seu trabalho ou caso queira, efectuar a operação de ressincronização dos seus dados a partir do servidor (sincronizando o perfil, as configurações e a *roaming briefcase*) ou para o servidor (sincronizando a *roaming briefcase*).

5.2.2 Operação Implícita do *Disconnected Mode*

Neste modo alternativo de operação o utilizador tem intervenção mínima, sendo o processo de incialização, manutenção e sincronização da *cache* levada a cabo de forma transparente. Para o efeito recorre-se a um *filesystem* distribuído com suporte para o paradigma do *disconnected computing* – baseado em Coda [Satyanarayanan 1989] – como ilustra a Figura 5.5.

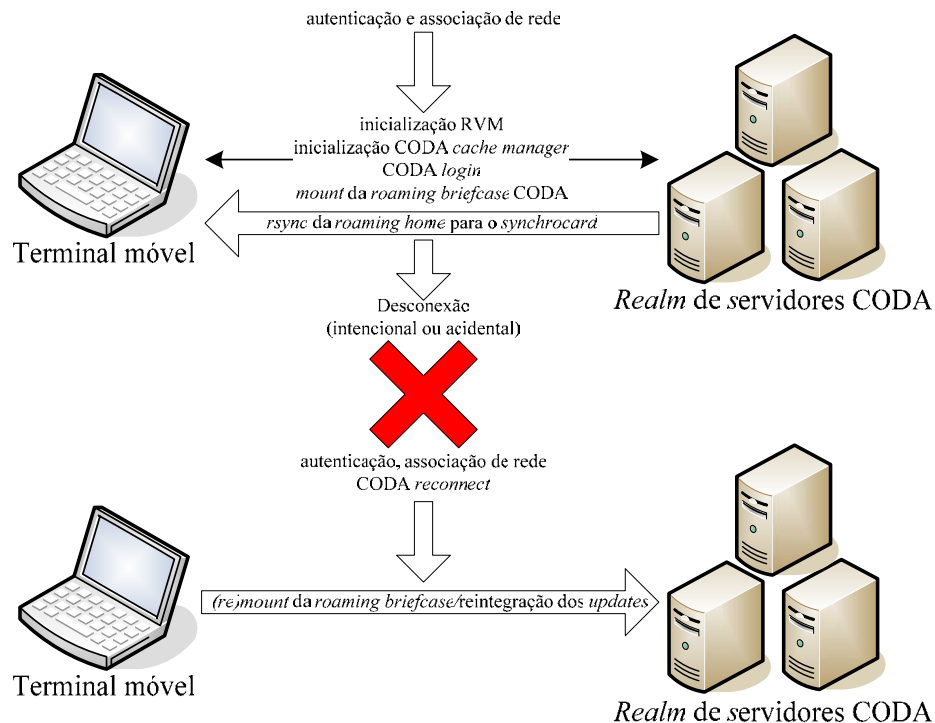



Figura 5.5 – Operação Implícita do *Disconnected Mode*

Depois do processo de autenticação e associação de rede, são iniciados os subsistemas RVM (*Recoverable Virtual Memory*) e *cache manager* Coda, seguindo-se autenticação do utilizador Coda no *realm* de servidores e o acesso à sua *roaming briefcase*. É depois efectuado o *rsync* da *roaming home* (constituída por dados de perfil de utilizador e configurações de programas e serviços) para o *synchrocard*.

Quando ocorre a desconexão o utilizador pode continuar a trabalhar nos ficheiros existentes na sua *roaming briefcase*, sendo mantido na *cache* RVM um *log* de modificações e operações efectuadas sobre estes ficheiros que permitirá actualizar a informação existente nos servidores do *realm*, uma vez restabelecida a conectividade de rede. O terminal móvel detecta automaticamente as falhas de rede e comuta de forma transparente para o modo *disconnected*

após um *timeout*, considerando-se nesta situação como *cacheable* todo o conteúdo da sua *roaming briefcase* [7].

Ainda que a gestão das falhas de conectividade seja implícita no exemplo da Figura 5.5, nada impede que a passagem para o modo *disconnected* possa ser feita manualmente, numa variante do modo explícito suportada com recurso ao Coda, podendo também o utilizador neste caso controlar a lista de ficheiros a manter em *cache* local aquando da desconexão.

A estratégia para lidar com a questão do *disconnected computing* descrita neste ponto é inovadora na medida em que é a primeira a recorrer a Coda para esses fins. Neste sentido subsistem algumas arestas por limar, nomeadamente no processo de autenticação e gestão de utilizadores Coda, que é independente do adoptado para os restantes serviços (baseados em RADIUS e serviço de directoria LDAP). O Coda recorre a mecanismos próprios de autenticação, obrigando a duplicar as contas dos utilizadores e a manter meios explícitos de sincronismo entre as duas infraestruturas de autenticação.

5.2.3 O *Filesystem* Distribuído Coda

O *filesystem* distribuído Coda – cuja principal característica é a inclusão de suporte para o *disconnected computing* – foi desenvolvido na Universidade Carnegie-Mellon [Coda]. O Coda surge como resposta a uma importante limitação de um *filesystem* distribuído anterior (o AFS, *Andrew File System* [Campbell 1998]), que apenas suportava replicação de volumes em modo *read-only*.

O Coda suporta *clustering* de volumes *read-write* distribuídos entre vários servidores (Figura 5.6), designados por *Volume Service Groups* (VSG). Os servidores de um VSG executam um processo dedicado à gestão das réplicas (*replica manager*), designado *Vice*.

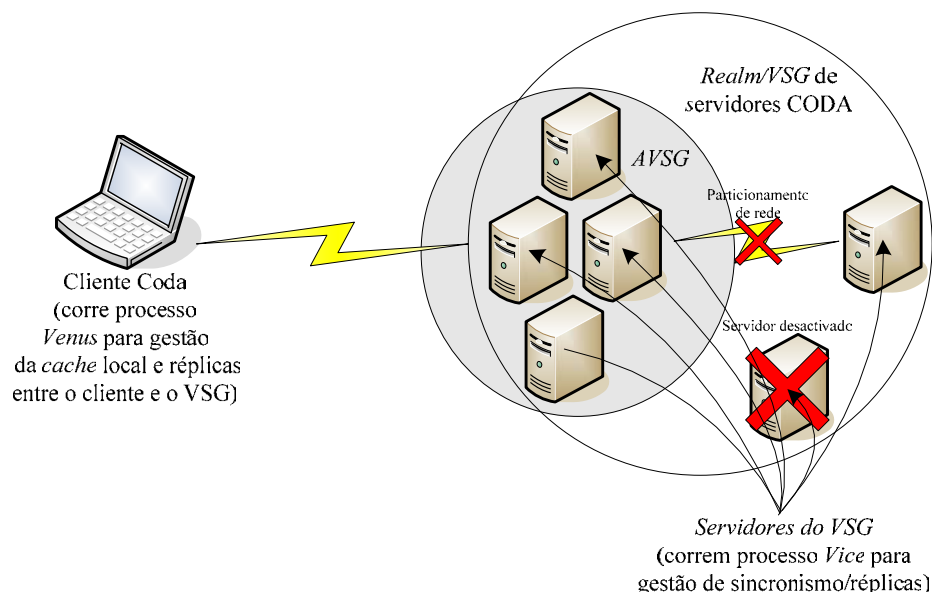



Figura 5.6 – *Clustering* de Volumes em Coda

[7] o Coda fornece um utilitário – o *spy* – que funciona em *background* com o propósito de inspeccionar as conexões entre o cliente e o servidor, identificando os ficheiros utilizados na sessão e que serão por conseguinte candidatos a incluir na *cache* persistente. Este método é eficaz para a maioria das situações, ainda que limite a flexibilidade do sistema na medida em que impede o utilizador de trabalhar em modo *disconnected* com ficheiros aos quais não tenha acedido antes da falha de conectividade de rede e que não foram, por conseguinte, eleitos para incorporação na *cache*.

O acesso dos clientes a um *Available Volume Service Group* (AVSG, que corresponde a um *subset* acessível de um VSG, podendo alguns servidores estar inacessíveis devido a particionamento de rede ou interrupção de serviço deliberada) é efectuado com o auxílio de um processo local designado *Venus*, que funciona como *front-end*, escondendo a implementação do *filesystem* das aplicações locais e permitindo que este opere de forma transparente em caso de falha de ligação de rede.

O processo *Venus* (Figura 5.7) é também um gestor de réplicas na medida em que gere os ficheiros existentes na *cache* persistente local, mantendo o sincronismo entre estes e as instâncias existentes no AVSG, gerindo a passagem ao modo *disconnected* em caso de falha, e gerindo o *journal* das operações a que estes foram submetidos – com o objectivo de, uma vez reestabelecida a ligação de rede, ressincronizar a informação pelos servidores do AVSG. A lista de ficheiros a manter em *cache* (*hoarded files*) pode ser gerada manualmente pelos utilizadores ou criada automaticamente com o auxílio do já mencionado processo *spy*.

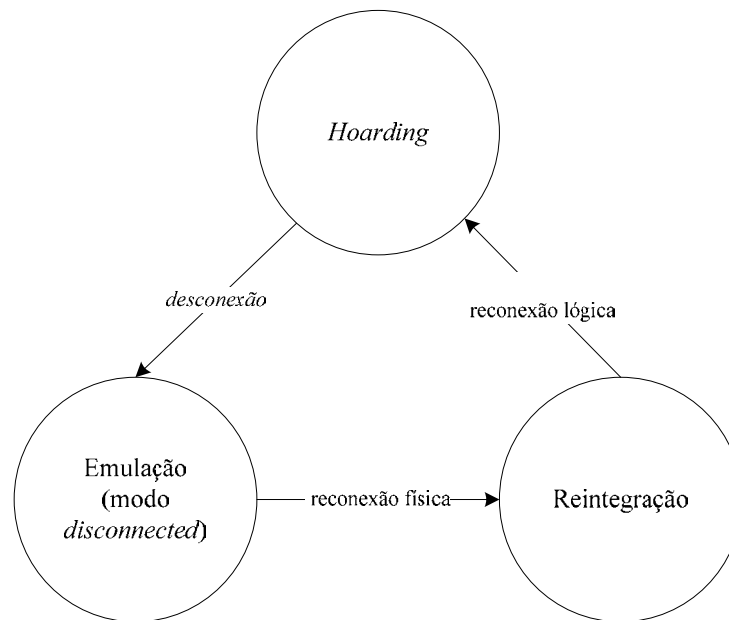



Figura 5.7: Funcionamento do Processo *Venus*

No *filesystem* Coda, quando a lista de membros de um AVSG se encontra vazia (o que implica que nenhum dos servidores do VSG está acessível), os clientes comutam para o modo *disconnected*, passando a utilizar os ficheiros existentes em *cache* local. Quando a conexão de rede é restabelecida os dados são difundidos pelos membros do AVSG[8].

A cada ficheiro é associado um *Coda Version Vector* (CVV) – um vector com uma *timestamp* por cada elemento do VSG onde o volume contendo o ficheiro se encontra. A função desta estrutura é manter informação sobre o historial de *updates* de cada réplica, para permitir a detecção de potenciais conflitos. O Coda admite, por concepção, a existência de dois tipos de conflitos:

- local/global (ou cliente/servidor). Este conflito surge quando um servidor detecta que o ficheiro que está a ser reintegrado não corresponde ao transferido originalmente para a máquina cliente, antes da desconexão;

[8] Note-se que, contrariamente ao que sugere alguma documentação, o Coda não utiliza *multicast* para este processo, o que limita a sua escalabilidade. Ao criar um VSG deverá pois ser imposto um limite razoável ao número de réplicas existentes, mantendo um compromisso entre redundância e desempenho

- servidor/servidor. Este conflito ocorre quando o sistema detecta que o ficheiro a reintegrar no AVSG não coloca problemas no servidor principal, existindo no entanto uma cópia noutro servidor da célula que é diferente daquela que o cliente trouxe.

Nestes casos a reintegração deverá ser feita manualmente, comparando as duas versões do ficheiro e decidindo qual deverá ser mantida no sistema. Para esse efeito o Coda possui um conjunto de ferramentas de suporte.

O acesso a réplicas num AVSG é feito com base numa variante do modelo *read-one/write-all*. Quando é desencadeada a abertura de um ficheiro não presente na *cache* local o cliente identifica um servidor preferencial no AVSG a partir do qual obtém uma cópia do ficheiro. A escolha deste servidor pode ser aleatória ou basear-se em critérios de desempenho, tais como a sua proximidade ou a sua carga. Quando é desencadeado o fecho de um ficheiro, o seu conteúdo e atributos são transmitidos em paralelo para todos os elementos do AVSG.

O *filesystem Coda* possui ainda outras características que o distinguem de outros *filesystems* distribuídos, tais como a possibilidade de adicionar ou suprimir dinamicamente servidores de um VSG e mecanismos de *backup* embebidos. É no entanto importante ter presente que, não obstante a importância das suas inovações, o Coda possui também um conjunto de limitações arquitecturais (*cfr. caixa*).

O *filesystem* Coda em Números: Limites

- **Nomes de volume**: 32 caracteres
- **Nomes de ficheiros**: 256 caracteres
- ***path names***: 1024 caracteres
- **ACLs por directoria**: 20
- **Tamanho máximo de uma directoria**: 256KByte
  (entre 2048 e 4096 ficheiros, devido ao tamanho médio do nome de ficheiro, *padding* e afins)
- **Volumes por servidor**: 1024
- **Réplicas por volume**: 8
- **Utilizadores e grupos**: até 2^31 (limite não testado experimentalmente)
- **Utilizadores por grupo**: até 2^31 (não testado testado experimentalmente)
- **Servidores por *realm***: 253 (limite 0-255, com os IDs 0, 127 e 255 reservados)
- **Tamanho máximo por ficheiro**:
  - limitado pelo tamanho da cache persistente – tentar fazer o *fetch* a um ficheiro maior que o tamanho da *cache* incorre em falha de acesso. Criar um ficheiro maior que a *cache* é possível, mas nunca se consegue obter de volta a partir dos servidores.
  - o cliente deve possuir espaço equivalente a 2 vezes o tamanho do ficheiro para qualquer instância que queira manipular em modo *disconnected*

5.2.4 Reforço da Fiabilidade e Flexibilidade do Modo *Disconnected*

O suporte para mobilidade na plataforma IC^3 é ainda um trabalho em progresso, existindo algumas lacunas, destacando-se especificamente: o *caching* de credenciais de autenticação; e a integridade do *synchrocard*.

O modo de operação *disconnected*, conforme está concebido e implementando, encontra-se direccionado para utilizadores de terminais móveis que trabalhem dentro dos limites físicos da rede de uma organização (zona de cobertura *wireless*), não sendo suportada a operação fora desse perímetro, pois todo e qualquer processo de autenticação requer acesso aos servidores pertinentes. Esta limitação foi inicialmente desvalorizada porque o primeiro objectivo do modo *disconnected* da plataforma IC^3 foi a tolerância a falhas de conectividade na rede *wireless* da organização. No entanto, não será complicado evoluir a plataforma de modo de modo a suportar mecanismos de *caching* local de credenciais de autenticação, conforme ilustra a Figura 5.8.

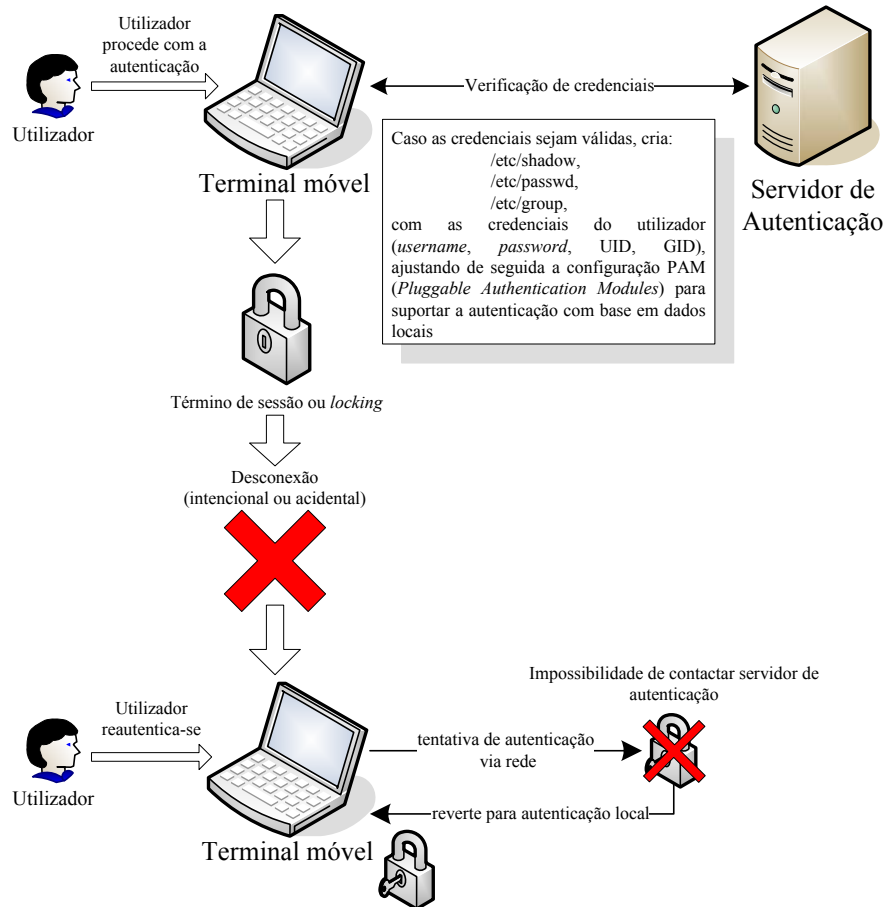



Figura 5.8 – *Caching* Local de Credenciais de Autenticação

No respeitante à integridade do *synchrocard* convirá referir que no protótipo actual não são tomadas quaisquer precauções para garantir a integridade do cartão ou dos dados nele armazenados, antes ou durante a sua utilização. Em futuras versões da plataforma deverão ser incorporados procedimentos de teste de *media* e controle de *wearout* (tais como contadores de utilização), para além dos tradicionais mecanismos de verificação de integridade do *filesystem*.

5.3 Infra-estrutura IC³ com Suporte para Mobilidade

A Figura 5.9 retoma a arquitectura de suporte preconizada para os postos de trabalho IC³ na secção 4.7.

Nesta perspectiva, os postos de trabalho móveis distinguem-se dos *wired* pelo suporte ao modo *disconnected*, incorporado com o objectivo de colmatar as limitações inerentes de conectividade de rede. De facto, se por um lado é viável suportar os postos de trabalho estáticos com recurso a conexões de rede persistentes – graças à fiabilidade da ligação baseada em cablagem – tal não é possível para os sistemas móveis, devido à menor fiabilidade das ligações *wireless*. Deste modo, optou-se pelo recurso a uma *cache* local armazenada num cartão amovível (o *synchrocard*), que tanto pode ser utilizada para armazenar a informação do utilizador de forma tradicional (operação em modo explícito) como pode servir de base para a operação de um *filesystem* com suporte para o modo *disconnected*, como é o caso do Coda. O recurso ao Coda permite que se montem as áreas dos utilizadores a partir dos servidores, da forma usual, mantendo e sincronizando as alterações efectuadas nos ficheiros quando existir ligação de rede activa.

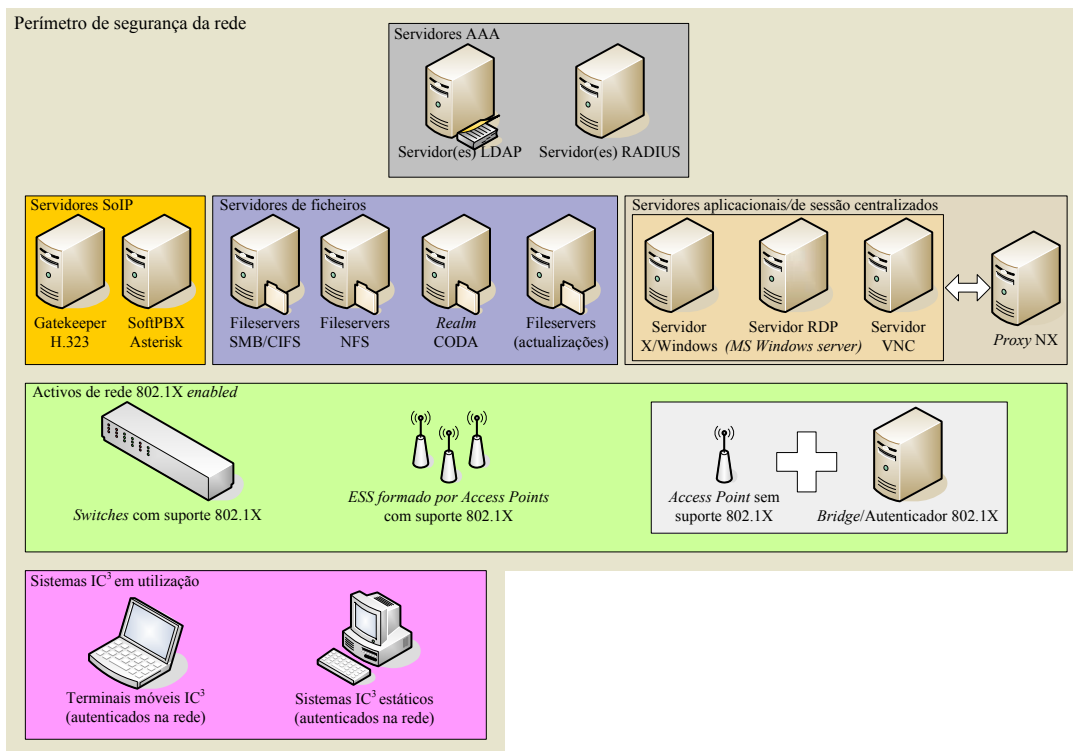



Figura 5.9 – Infraestrutura da Plataforma IC³ com Suporte de Mobilidade

Nos casos em que a ligação falha ou é suspensa o Coda, ao invés dos *filesystems* tradicionais, permite que o funcionamento das aplicações continue de forma normal e transparente, operando sobre as instâncias dos ficheiros existentes na *cache* local e inciando um registo de transações para cada ficheiro que será utilizado para actualizar a informação nos servidores, uma vez reestabelecida a conectividade. Deste modo o utilizador pode continuar o seu trabalho, apesar da interrupção da ligação de rede que o mantinha conectado ao servidor onde se encontrava o ficheiro com que estava a trabalhar no instante da falha.

6. Conclusão

Neste Capítulo, que finaliza a Dissertação, apresenta-se uma síntese do trabalho efectuado, identificam-se as suas principais contribuições e apontam-se linhas de trabalho futuro.

6.1 Síntese

Revisitando a panóplia de paradigmas de utilização dos meios informáticos – que foram acompanhando a evolução das próprias plataformas computacionais – torna-se claro que cada um desses paradigmas possui um conjunto de vantagens e deficências inerentes à sua própria natureza. Desde os sistemas centralizados até à era do modelo cliente-servidor cada paradigma procura colmatar as lacunas da geração anterior, fazendo o melhor uso possível da tecnologia disponível no sentido de maximizar a produtividade dos utilizadores, num processo onde muitas vezes as questões relacionadas com a gestão e o custo de operação passam desapercebidas, sendo apenas tratadas *a posteriori*. Foi precisamente o que sucedeu com o PC tradicional e com as suas lacunas em termos de gestão, derivadas da natureza da plataforma de *hardware* em que assenta e da descentralização do poder computacional pelos vários postos de trabalho da organização, geralmente desprovidos de meios eficazes de controlo e/ou diagnóstico.

Surgiram gradualmente tecnologias e normas pensados especificamente para colmatar essas lacunas, ao mesmo tempo que se tentaram operar modificações ao próprio PC enquanto plataforma. No entanto, a maior parte destas iniciativas não abordaram a problemática do gestão do *desktop* numa perspectiva global.

O projecto IC^3, enquanto continuação lógica do trabalho anteriormente desenvolvido no LCT do DEI/FCTUC na vertente de gestão de *desktops*, procurou tratar a questão recorrendo a um paradigma alternativo: o conceito de *balanced computing*. Procurando encaixar-se no espaço disponível entre o PC tradicional e os paradigmas do *network-computing* e *thin-clients* (de natureza centralizada), este paradigma pretende obter um melhor equilíbrio entre o uso dos recursos computacionais locais do posto de trabalho (que, no caso dos sistemas centralizados, não existem ou não são simplesmente utilizados) e os recursos da infraestrutrura de servidores.

Incorporando ainda a noção de convergência de serviços de comunicação e computacionais, sob a forma do suporte para um conjunto de Serviços sobre IP, nomeadamente VoIP e Videoconferência, a plataforma prevista no projecto IC^3 pretendeu demonstar a viabilidade do conceito de *balanced computing* com base num sistema eficiente do ponto de vista energético e computacional, sem estado local persistente, e capaz de servir *per se* as necessidades do posto de trabalho típico. A ausência de estado local, associada a mecanismos de directoria, autenticação e localização apropriados, permite que os utilizadores possam trabalhar em regime de *roaming*, sem a necessidade de postos de trabalho físicos fixos, sendo o utilizador "seguido" pelo seu ambiente de trabalho e preferências de utilizador.

Aos benefícios previstos para os postos de trabalho fixos (*wired*), o projecto IC^3 acresce ainda o suporte para postos de trabalho móveis (concebido para postos de trabalho que estejam ligados à infraestrutura de rede corporativa por meio da sua rede *wireless*), com incorporação de suporte para *disconnected computing*.

6.2 Contribuições

Em balanço final, considera-se que a maior realização deste projecto terá sido a concretização de uma prova do conceito no protótipo da plataforma IC^3, constituído por dois postos de trabalho fixos, por um posto de trabalho móvel e pelos diversos servidores de suporte da infraestrutura.

Para além da validação do conceito, este protótipo mostrou também que é possível recorrer a *software* e normas abertos e divulgados para construir sistemas que equilibrem as exigências de flexibilidade de manutenção e gestão com a liberdade concedida aos utilizadores.

Dos diversos aspectos relacionados com a concepção e desenvolvimento da plataforma, destaca-se o seguinte subconjunto:

- **Disponibilidade.** Tanto a plataforma a desenvolver como a arquitectura de suporte foram concebidas de modo a fornecer um serviço de elevada disponibilidade, com elevado grau de resiliência e robustez a falhas, sejam elas induzidas ou acidentais. Para este efeito, foi tomada especial precaução com os factores de redundância e fiabilidade, além da adopção do suporte para o *disconnected computing*.
- **Eficiência e Equilíbrio.** A eficiência, seja ela em termos energéticos, de gestão, arquitecturais ou computacionais, foi uma das características às quais foi concedida mais atenção no projecto IC^3. Desde o consumo de energia da versão *wired* da plataforma à concepção realizada em torno de uma arquitectura sem estado local, procurou-se atingir um equilíbrio adequado entre as necessidades/problemas a endereçar e os recursos considerados necessários para que a arquitectura a desenvolver, sem contudo sacrificar a facilidade de utilização e liberdade dos utilizadores.
- **Simplicidade.** Ainda que a solução desenvolvida possa parecer complexa, quer do ponto de vista dos postos de trabalho quer do ponto de vista da infraestrutura de suporte, a verdade é se mantém uma assinalável simplicidade. A panóplia de tecnologias e servidores de suporte já se encontra presente na maior parte das redes corporativas (bastando ligeiras afinações na sua configuração) e o posto de trabalho em si é bastante mais simples que um PC tradicional.
- **Custo.** A solução desenvolvida responde à problemática do TCO atacando um amplo leque de pontos críticos, desde o tempo necessário para substituir, reparação e reinstalação postos de trabalho em caso de falha até à consolidação dos meios de armazenamento de massa.

A satisfação destes requisitos específicos permitiu concretizar o *balanced computing* da forma inicialmente idealizada. Com a introdução deste paradigma, a infraestrutura informática de uma organização poderá ganhar em termos de plasticidade, adoptando modos de organização do trabalho mais evoluídos, menos restritivos e com benefícios para os utilizadores.

6.3 Trabalho futuro

A solução aqui apresentada carece ainda de trabalho adicional, de modo a colmatar algumas lacunas e fragilidades. Destaca-se:

- O reforço da robustez do 802.1X em ambientes *wired* e o estudo de alternativas para a criação e manutenção de perímetro seguro (cfr. Secção 4.5.3).
- a escolha de um *filesystem* distribuído para os postos de trabalho fixos com suporte adequado para redundância (cfr. Secção 4.5.4).
- a duplicação de informação de autenticação entre a infraestrutura de serviços e o SoftPBX (cfr. Secção 4.5.6).
- no funcionamento em modo *disconnected,* as questões relacionadas com o *caching* de credenciais e a verificação de integridade do *synchrocard* (cfr. Secção 5.2.3).

Estes foram os principais pontos identificados como carecendo de trabalho suplementar – e cujo âmbito caiu fora desta dissertação – de modo a melhorar a plataforma IC^3.

Por último, fica por realizar – por manifesta falta de recursos – um trabalho mais extenso de avaliação da plataforma. Ainda que o protótipo criado permita aferir, de forma subjectiva, a viabilidade e potencialidade da plataforma – os postos de trabalho apresentam um desempenho adequado, os mecanismos de resiliência e robustez funcionam da forma prevista, o impacto da plataforma na rede de comunicações está dentro dos parâmetros esperados, a administração e manutenção da infraestrutura é funcional e eficaz – será naturalmente

necessário proceder, futuramente, a uma avaliação mais extensa, sistemática e quantitativa da plataforma, ao nível do impacto nos utilizadores, do TCO, da segurança, da resiliência e da funcionalidade.

A. Mecanismo de Actualizações

Neste anexo discutem-se os protocolos utilizados para descarga das imagens do sistema via rede – para efeitos de actualização do ambiente de operação IC^3 – tendo em conta as tecnologias de rede suportadas e o binómio custo/benefício decorrente de cada aproximação. Procura-se assim fazer uma avaliação do conjunto de possíveis soluções – baseadas em protocolos *unicast, broadcast* e *multicast* – enquadrando cada uma no contexto específico deste projecto.

Inclui-se também um breve resumo sobre as tecnologias de *reliable multicast* existentes, seja o seu modelo de operação ASM – *Any Source Multicast* – ou SSM – *Source Specific Multicast*.

A.1 Soluções Baseadas em Protocolos *Unicast*

De acordo com a Secção 4.5.1, o método mais conservador suportado pela plataforma IC³ para descarga da imagem de sistema consiste no recurso aos protocolos FTP ou SFTP (o *rsync* [Rsync] ou o NFS [RFC1813] poderiam também ser usados com os mesmos resultados), de natureza *unicast* (Figura A.1). Dado que cada transferência é realizada com recurso a um fluxo de dados ponto-a-ponto, e que existe a necessidade de gerar *n* fluxos concorrentes para suportar a distribuição para *n* clientes, o servidor deve disponibilizar o débito necessário para suportar os fluxos concorrentes (quer em termos de interface de rede, quer em termos dos subsistemas de I/O internos) e a infraestrutura de rede deve estar igualmente dimensionada para os suportar (pode-se mesmo afirmar que este modelo é fundamentalmente *I/O bounded*).

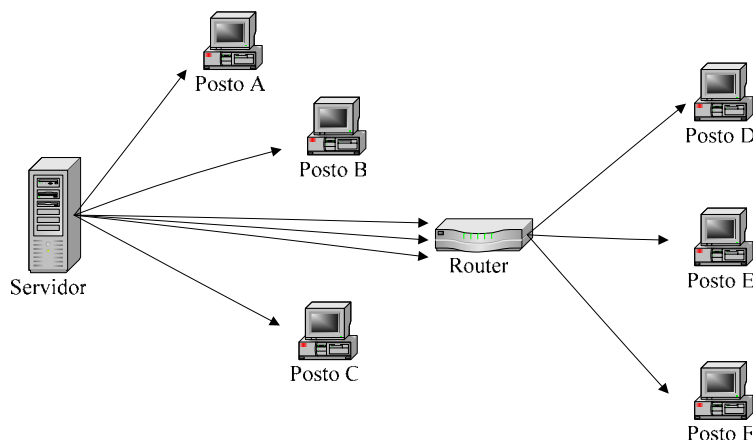
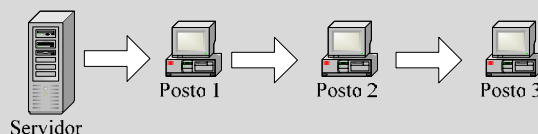



Figura A.1 – Descarga de Imagens do Sistema por Protocolos *Unicast*

Com um tamanho de cerca de 410MByte, no protótipo actual, a actualização do ficheiro contendo a imagem de sistema IC³ (cfr. Secção 4.4.2) pode constituir um sério problema, à medida que aumenta o número de postos de trabalho. Ainda que existam alternativas eficientes baseadas em protocolos *unicast* (*cfr. caixa*), estas são inadequadas quando o ambiente não é constituído exclusivamente por sistemas *wired ethernet*.

Alternativas Eficientes sobre Protocolos *Unicast*

Uma alternativa para distribuição de ficheiros em *unicast* consiste na distribuição em cadeia. Um bom exemplo do uso desta técnica pode ser encontrado na ferramentas *Dolly* [Dolly] e *Dolly+* [Dolly+] desenvolvidas no contexto do *Projecto Patagónia* do *ETH Zürich* [Patagonia]. O funcionamento destas ferramentas baseia-se na criação de uma cadeia TCP virtual que interconecta as máquinas que vão receber e reenviar o fluxo de dados.

Distribuição em cadeia

Torna-se evidente que esta aproximação potencia as vantagens das redes comutadas (nomeadamente redes *switched ethernet*), envolvendo um fluxo de dados único à saída do servidor. Ainda que cada cliente passe a suportar dois fluxos de dados (*upstream* e *downstream*), este método permite resultados muito interessantes (2 GByte replicados por 15 postos de trabalho em menos de 4 minutos, numa rede *gigabit ethernet*), utilizando para o efeito alguns *scripts* bastante simples. Como se recorre ao uso do TCP, esta técnica garante a integridade dos dados. O seu ponto mais fraco reside na possibilidade de falha de um dos nós intermédios. Contudo, esta fragilidade pode ser resolvida pelos mecanismos de *bypass* com *timeout* implementados no *Dolly+*).

Estes métodos são intrinsecamente inadequados para ligações em meio partilhado, como é o caso das redes 802.11, onde poderiam provocar sérios problemas de ocupação de canal.

A.2 Soluções Baseadas em Protocolos *Broadcast*

Existem também alternativas que têm por base o *broadcast* (Figura A.2)*,* como é o caso do CFDP (*Coherent File Distribution Protocol*) [RFC1235]. Estas soluções têm em comum o facto de funcionar graças à difusão de pacotes IP (envio para o endereço de *broadcast* da rede), reduzindo o débito necessário à saída do servidor para um único fluxo. Contudo, todos os nós recebem o tráfego, independentemente de lhe interessar ou não, e os fluxos não são encaminhados pelos *routers*, pelo que os nós existentes a jusante destes não podem receber o tráfego. Existe ainda outra limitação: os pacotes recebidos são processados pelo núcleo do sistema operativo/pilha protocolar, implicando um *overhead* desnecessário nos casos em que o pacote é recebido por uma estação que não o deseje.

Estes inconvenientes são relevantes mesmo no contexto dos sistemas *wired*, assumindo a sua dimensão maior magnitude em ambientes *wireless* (*shared medium*, *half duplex,* menor débito, maiores latências), tornando-os desinteressantes no contexto do IC^3.

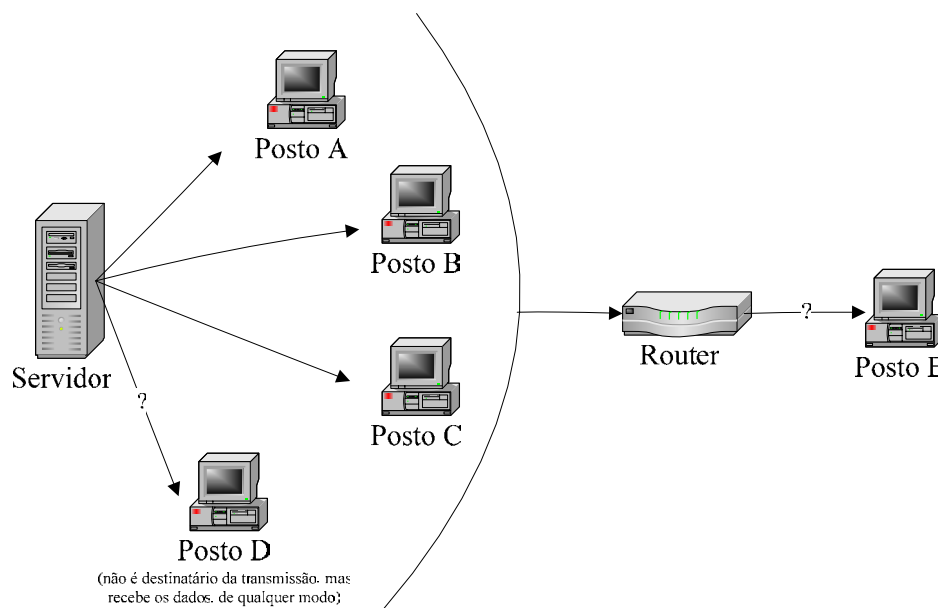



Figura A.2 – Descarga de Imagens do Sistema por Mecanismos *Broadcast*

A.3 Soluções Baseadas em Protocolos *Multicast*

É portanto desejável dispor de um método que permita distribuir ficheiros de forma selectiva, utilizando de forma racional a largura de banda disponível. Tal é possível usando transmissão *multicast*, sendo os fluxos de dados recebidos apenas apenas pelos nós pertencentes a um dado grupo (Figura A.3).

Este método permite que o servidor disponibilize um fluxo único de dados à saída (análogo ao *broadcast* e *unicast* em cadeia), recebido apenas pelos subscritores do grupo a que este se destina. Se for necessário que este seja encaminhado por um *router* (devidamente configurado e conforme com o nível 2 da especificação *multicast*) para atingir um nó a jusante, tal será feito com um único fluxo entre as redes envolvidas. Outra das grandes vantagens do *multicast* reside no facto de este ser naturalmente suportado pela norma *Ethernet*, pelo que a filtragem de pacotes é efectuada a nível do *transceiver* da placa de rede (com excepção de alguns adaptadores mais antigos, que apenas suportam *multicast* utilizando o modo promíscuo de operação), não constituindo *overhead* para os restantes clientes.

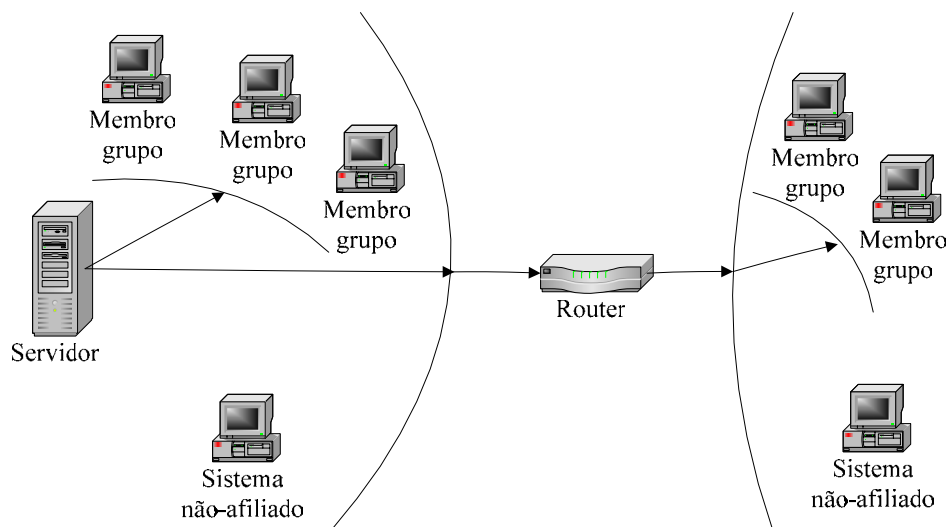



Figura A.3 – Descarga de Imagens do Sistema por Mecanismos *Multicast*

Existe contudo uma desvantagem no uso de *multicast* – mais especificamente do *multicast IP/UDP*. Devido às características intrisecas do protocolo UDP, são necessários mecanismos suplementares que assegurem a integridade dos dados recebidos. Nesta óptica, os protocolos para transferência de dados em *multicast* podem ser divididos em duas famílias: os baseados em *feedback* e os de *feedback zero*.

Para assegurar a integridade da informação transmitida, as soluções baseadas em *feedback* utilizam, como o próprio nome indica, um mecanismo de retorno de informação que permite confirmar se a transmissão correu como esperado ou, em alternativa, pedir a retransmissão dos pacotes em falta. Estes métodos dividem-se em duas categorias: os de confirmação síncrona (*positive acknowledge*) e os baseados em repetição de transmissão a pedido (NACK- *negative acknowledge* ou ARQ – *Automatic Repeat Request*).

O *positive acknowledge* (Figura A.4) funciona com base no princípio da confirmação de cada pacote recebido. Este método não é prático para grupos de grandes dimensões devido ao efeito secundário da implosão do *feedback* recebido pelo servidor (*ACK storm*), que limita a escalabilidade.

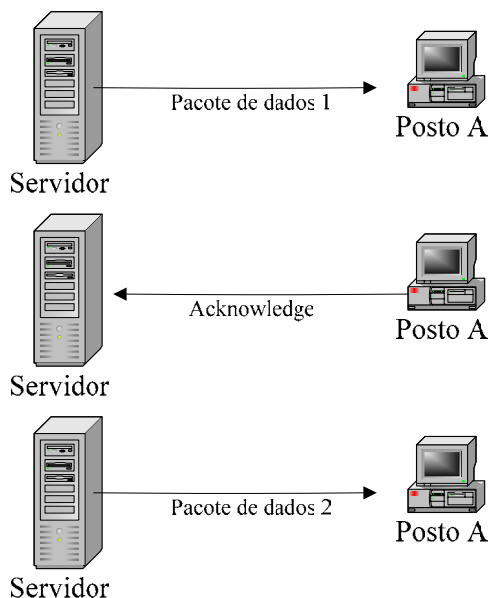



Figura A.4 – *Multicast* com *Positive Acknowledge*

Apesar deste inconveniente, o *positive acknowledge* pode ser tolerado em ambientes de rede local, para grupos de dimensão moderada. Além disso, a simplicidade da sua implementação permite que seja embebível em *Boot ROMs* e outras aplicações onde o espaço disponível para código e dados não abunde.

Em alternativa, é possível minorar o impacto das *ACK storms* recorrendo a mecanismos de propagação hierárquica do *feedback* com supressão, nos quais apenas passa uma resposta de um nível para o seguinte (suportado ao nível dos *routers*).

Outra hipótese consiste em, ao invés de confirmar todos os pacotes, emitir apenas um pedido para retransmissão dos pacotes perdidos – *negative ACK* (Figura A.5). Neste caso, a quantidade de mensagens de retorno é francamente diminuída, se bem que não completamente eliminada, pois a probabilidade de ocorrer uma (*N)ACK storm* mantém-se. Basta que um grupo isolado atrás de um *router* seja vítima de um particionamento momentâneo da rede, em processo de recepção de um fluxo de dados, para que se gere uma grande quantidade de pedidos de retransmissão (implosão de retorno).

Contudo, existem esquemas probabilísticos baseados na retransmissão do pedido após um intervalo de tempo aleatório que permitem minimizar a ocorrência de tais eventos, tornando o ARQ/NACK viável para uso em redes com probabilidade de perda de dados heterogénea.

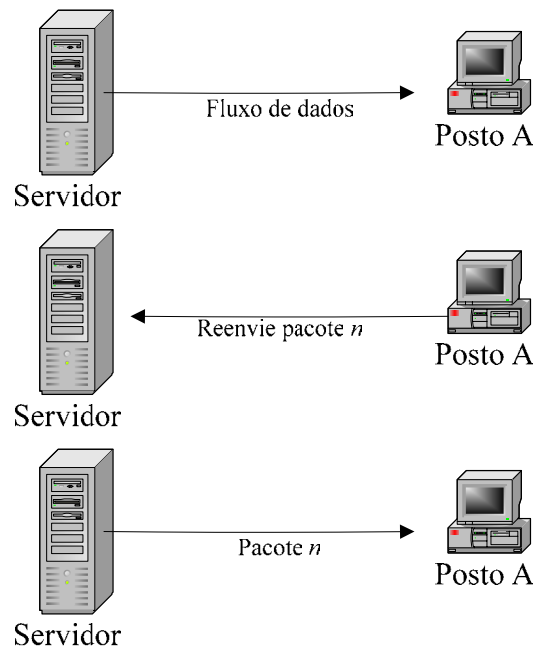
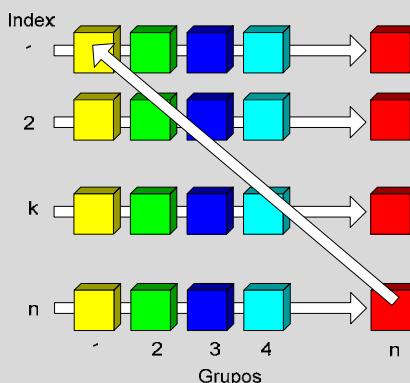
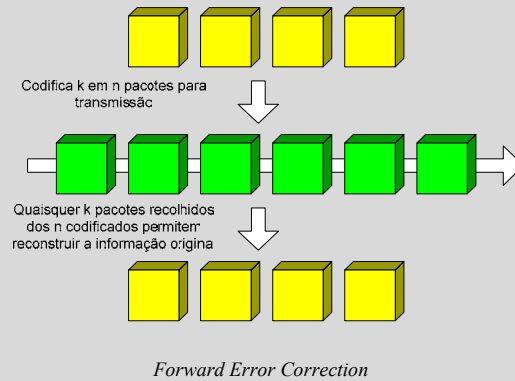



Figura A.5 – *Multicast* com *Negative Acknowledge*

Em situações em que o *feedback* dos clientes não é de todo desejável, seja por não existirem canais adequados (caso de uma sonda espacial), seja porque o *round-trip* time é elevado (caso de um satélite), ou seja ainda porque se quer eliminar de todo a possibilidade de uma *(N)ACK storm*, recorre-se a mecanismos de *Forward Error Correction* [Aerospace] que permitem embeber no fluxo de dados a informação necessária para recuperação de perdas, combinados com técnicas de retransmissão automática (*cfr. caixa na página seguinte*).

Códigos *Forward Error Correction* (FEC) e Protocolos *zero feedback*

Os mecanismos do tipo FEC fazem uso de códigos adequados para recuperação de erros, como os do tipo *Reed--Solomon* [Reed 1960], para embeber nas mensagens informação redundante que permita detectar a perda de pacotes. Como as camadas inferiores da pilha protocolar TCP asseguram a verificação da integridade dos dados, o uso do FEC no contexto da distribuição de ficheiros destina-se a detectar e recuperar de perdas de pacotes quando se recorre a protocolos *connectionless* baseados em UDP.

Forward Error Correction

Cada conjunto de *k* pacotes é codificado gerando *n* pacotes (utilizando, por exemplo, a codificação Reed-Solomon). Esta codificação (*n*,*k*) possui uma propriedade que lhe permite recuperar os *k* pacotes originais a partir de qualquer conjunto de *k* indivíduos da população dos *n* pacotes resultantes da codificação efectuada.

Contudo, o FEC não oferece garantia absoluta da fiabilidade da recepção: basta receber apenas *y* pacotes (com $y<k$) para que seja impossível recuperar a mensagem. Nestes casos há duas hipóteses: ou se recorre a mecanismos de ARQ (reintroduzindo o *feedback* e a possibilidade de *feedback implosion/NACK storm*), ou opta--se pela técnica do *data carrousel,* baseada na retransmissão cíclica dos dados ao longo do tempo por forma a que um falha crítica do FEC possa ser recuperada pela subsequente transmissão da mesma informação.

Para aligeirar o peso computacional do algoritmo FEC recorre-se à divisão dos dados em grupos, que por sua vez serão submetidos a codificação sendo enviado um bloco de cada grupo alternadamente para minimizar o tempo de recepção – *interleaved FEC*. De facto, nada obriga a que a sequência de transmissão seja a mesma a cada ciclo: ao enviar alternadamente os dados, evita-se a situação em que para um ficheiro de *g* pacotes o receptor tenha de esperar um ciclo inteiro para recuperar apenas um pacote perdido.

Interleaved FEC + *Data Carrousel*

Alguns protocolos utilizados correntemente para difusão *multicast* de ficheiros recorrem a esta técnica. Um bom exemplo é o *Fcast* da *Microsoft* [Gemmell] – desenvolvido na sequência da situação caótica gerada aquando da disponibilização para *download*, alguns anos atrás, do *browser Internet Explorer 3.0*, que levou ao colapso dos *web servers* desta empresa – e o FLUTE/FCAST do INRIA [INRIA].

A.4 Protocolos *multicast* e Operação em Ambientes *Wireless*

No contexto do IC^3, o mecanismo mais adequado para distribuição das imagens com as actualizações do ambiente de sistema será, sem dúvida, baseado em *multicast*. Contudo, há que ter mais uma vez em consideração o facto de o ambiente de operação previsto contemplar redes *wired* e *wireless*.

Devido à sua maturidade e difusão, certos aspectos relativos às redes *Ethernet wired* são sobejamente conhecidos, divulgados e documentados, não sucedendo o mesmo na tecnologia 802.11 *wireless*, o que em parte se deve à sua relativa juventude. Assim, considerou-se pertinente (e prudente) ressalvar os seguintes aspectos relativos a esta tecnologia que influenciaram de forma decisiva as opções tomadas:

- As redes 802.11 possuem débitos reais mais reduzidos, devido à natureza do meio utilizado e ao maior *overhead* derivado da informação extra que circula para recuperação de erros, protecção da *payload*, controlo de tráfego e gestão. É comum para uma rede 802.11b (11Mbps) obter débitos reais inferiores a 5Mbps ou 6Mbps.
- As redes 802.11 operam em modo *half-duplex.* Para solucionar esta situação seria necessário mais do que um *radio transceiver*.
- O tráfego que circula em redes 802.11 está sujeito a maior latência.
- As redes 802.11 possuem suporte para *multicast* de tramas, também designado *group frames*. Contudo, este suporte é *unacklowledged*, não existindo mecanismos de confirmação de recepção (*cfr. caixa na página seguinte*). Além disso, estas *frames* não podem ser fragmentadas, o que aumenta a sua vulnerabilidade ao ruído.
- Os pacotes *multicast* são transmitidos com o mesmo débito da estação com a conexão mais lenta que esteja associada a um AP, mesmo que este suporte velocidades superiores.
- O suporte *multicast IP* é implementado sobre as *frames multicast* do protocolo 802.11.

Estas características, específicas das redes 802.11, exigem alguma precaução na escolha do mecanismo de difusão de actualizações, de modo a tirar o melhor proveito possível das conexões existentes, conservando largura de banda. Nesta situação é desaconselhado o recurso a mecanismos com *feedback* intensivo ou elevada probabilidade de gerarem *traffic storms* (*ACK implosions*).

A.5 Protocolos *Multicast* Avaliados

Deste modo, consideraram-se várias alternativas para a difusão/actualização via *multicast*:

- O **FLUTE** (*File Delivery over Unidirectional Transport*) e o **FCAST** (a não confundir com o FCAST da Microsoft). As versões destes protocolos que foram testadas são baseadas na MCL v3 (*MultiCast Library* [INRIA]) do INRIA.

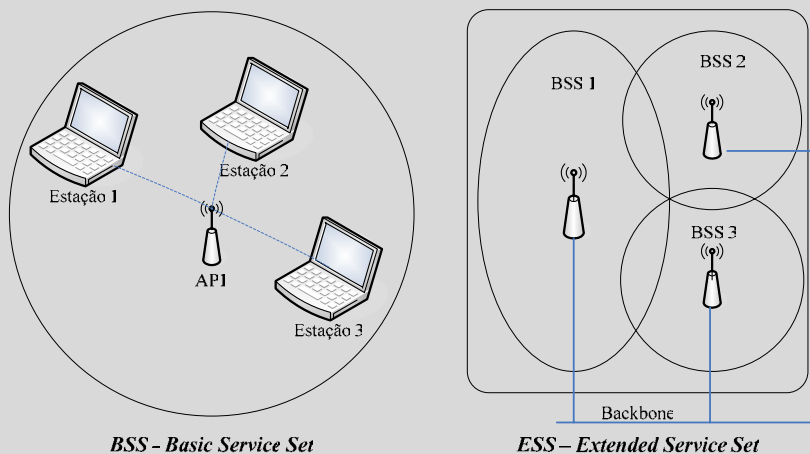
  Estes dois protocolos são semelhantes em termos de conceito e modo de operação. O FLUTE é uma implementação do [RFC3926], com recurso ao protocolo ALC (*Asynchronous Layered Coding*) [RFC3450], operando sob o pressuposto da existência de conectividade entre o emissor e o receptor. O FCAST suporta igualmente a operação em modo ALC, se bem que não seja conforme com o RFC3926, ao que acresce a possibilidade de operar com o protocolo NORM (*Negative Acklowledgment Oriented Reliable Multicast*), um protocolo clássico de tipo NAK.

Multicast e Mecanismos ARQ: *wired vs. wireless*

Nas redes *ethernet wired* é o próprio emissor que detecta a colisão de um pacote e procede com a sua retransmissão (mecanismo ARQ). Contudo, nas redes *wireless 802.11*, a detecção de colisões (a principal causa de perca de pacotes *multicast*) é mais complexa. A política de acesso ao meio nas redes 802.11 é baseada no método CSMA/CA (*Carrier Sense Multiple Access* / *Colision Avoidance*), por oposição ao CSMA/CD (*Carrier Sense Multiple Access* / *Colision Detect*) da *wired Ethernet*.

As colisões podem surgir pelos mais diversos motivos, nomeadamente a colisão entre APs (*Access Points*) em BSS (*Basic Service Set*: um conjunto formado por um AP e uma ou mais estações) sobrepostos no mesmo canal e no mesmo ESS (*Extended Service Set*: um conjunto de BSS interligados por um sistema de distribuição, que fornecem mobilidade a nível da camada de ligação de forma uniforme entre si) ou a colisão com pacotes *unicast* provenientes de estações no mesmo canal do AP. Estes comportamentos são especialmente hostis para o *multicast* devido a duas situações:

- Num BSS, as estações estão explicitamente impedidas de vigiar APs aos quais não estejam associadas. Assim, quando um emissor *unicast* envia um pacote *unicast* existe uma forte possibilidade de este colidir com um pacote *multicast* enviado por um AP.
- Um pacote *multicast*, quando enviado, deve ser retransmitido por todos os APs num ESS, aumentando o risco de colisões.

BSS - Basic Service Set ***ESS – Extended Service Set***

Além disso, o meio *wireless* é bastante vulnerável a ruído. O BER (*Bit Error Rate*) estimado do 802.11 é de cerca de 10^{-5} [Cisco], bastante acima do BER típico de rede s *wired* (10^{-10}).

Assim sendo, as WLAN recorrem a um mecanismo de *acknowledgments* implementado na camada física, que não contempla as *frames multicast* e que opera com recurso à confirmação (ACK) por cada pacote recebido, permitindo inferir se uma dada estação recebeu ou não a informação transmitida (se um ACK não foi recebido, deduz-se que o pacote se perdeu). Este processo permite detectar perdas de forma satisfatória em *streams unicast*. Contudo, uma das condições da implementação do *multicast* é a possibilidade de uma estação poder abandonar o grupo inadvertidamente sem informar o emissor, o que invalida o seu uso nestas situações. Por este motivo, não existe nenhum mecanismo de ACK a nível do MAC (*Medium Access Controller*) em 802.11 *multicast* que permita a retransmissão de tramas perdidas, o que resulta em menor qualidade de serviço, agravada ainda pela ausência de ACKs, que impede a implementação de mecanismos de *backoff*.

O ALC (Figura A.6), que é o denominador comum entre o FLUTE e o FCAST, opera com base em códigos FEC em que o controlo de congestão é conseguido enviando pacotes da sessão para os vários grupos envolvidos, podendo os receptores fazer variar o seu débito de recepção de sessão, aderindo ou deixando os grupos ALC associados com esta, de modo *network friendly*. Tal é conseguido devido à natureza orientada ao receptor do protocolo, em que o emissor envia informação de controlo

junto com o tráfego transmitido indicando aos receptores como devem reagir e ajustar o seu débito de recepção em função do *packet loss*.

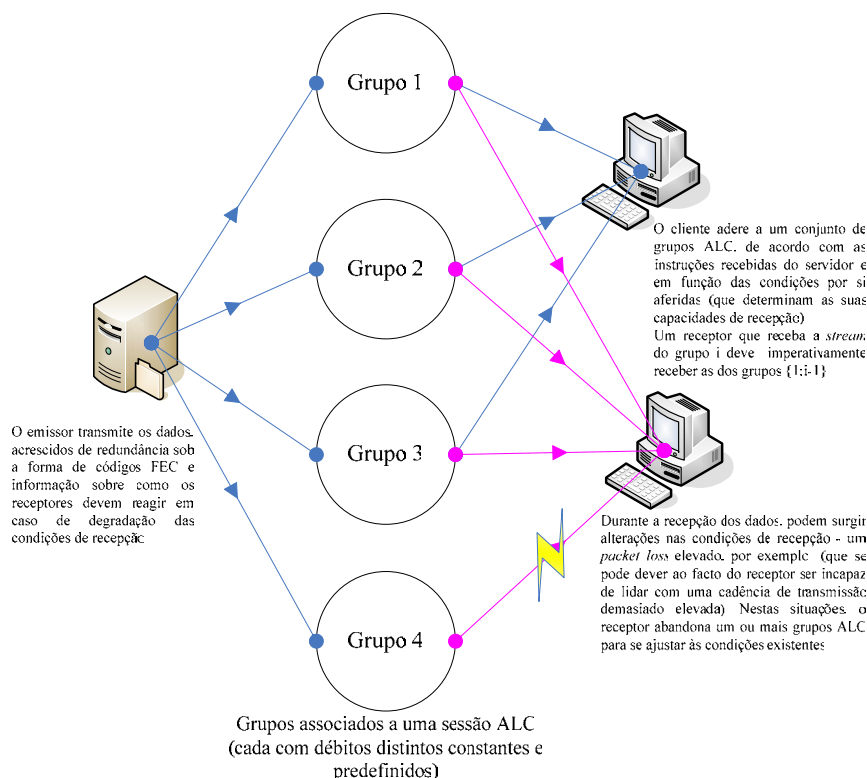



Figura A.6 – Funcionamento do ALC (*Asynchronous Layered Coding*)

Infelizmente, os testes efectuados com duas implementações do ALC (FCAST e FLUTE do INRIA) demonstraram existir um problema derivado do *setup time* ser demasiado elevado, podendo mesmo atingir-se vários minutos de espera numa LAN antes que um cliente comece a descarregar informação. Adicionalmente, ambas as implementações aparentam alguma imaturidade. O facto de recorrer a um *data carrousel* (modelo *on-demand delivery* em que os clientes podem chegar a qualquer momento e onde os dados são enviados continuamente) também impõe *stress* na conexão indepentemente da existência de clientes, o que não é satisfatório para ambientes *wireless*.

É ainda de referir que na família dos protocolos ALC existem ainda as variantes *Swarmcast* [OnionNetworks] e JRMS [Rosensweig 1998], cuja dependência da plataforma Java as coloca fora de questão para o propósito do sistema de *updates* desenvolvido, devido ao *overhead* do suporte para a JVM no ambiente encapsulado criado para lidar com o processo.

- O **PXE-MTFTP** foi outro protocolo considerado. Este protocolo foi desenvolvido pela *Intel* para fazer parte da especificação PXE (*Preboot eXecution Environment,* [Intel 1999b]) e é baseado num *internet draft* de 1997 [RFC2090]. Para permitir a transferência de ficheiros de forma racional e eficiente, durante a operação da *boot ROM PXE*, a *Intel* optou por arquitectar uma variante do TFTP (*Trivial File Transfer Protocol* [RFC1350]) baseada em *IP multicast*: o MTFTP (*Multicast Trivial File Transfer Protocol* [Intel 1999b]).

Devido a constragimentos físicos dos componentes de *firmware* da arquitectura PC, houve a necessidade de incorporar fiabilidade no TFTP minimizando o uso de espaço em ROM/Flash, o que determinou o recurso a mecanismos de *feedback*.

Resumidamente, o MTFTP funciona da seguinte forma:

1 - Numa fase incial o cliente monitoriza o tráfego do grupo a que pertence e se encontrar o fluxo que lhe interessa recolhe os pacotes, sem fazer ACK.
2 - Se na primeira fase não tiver recebido todos os pacotes e não for detectado nenhum fluxo pertinente em circulação, o cliente emite um pedido de transferência dos pacotes em modo *unicast*.
3 - Em modo *unicast* o cliente confirma todos os pacotes recebidos, incluíndo os pacotes relativos à fase de recepção *multicast* (Figura A.7)

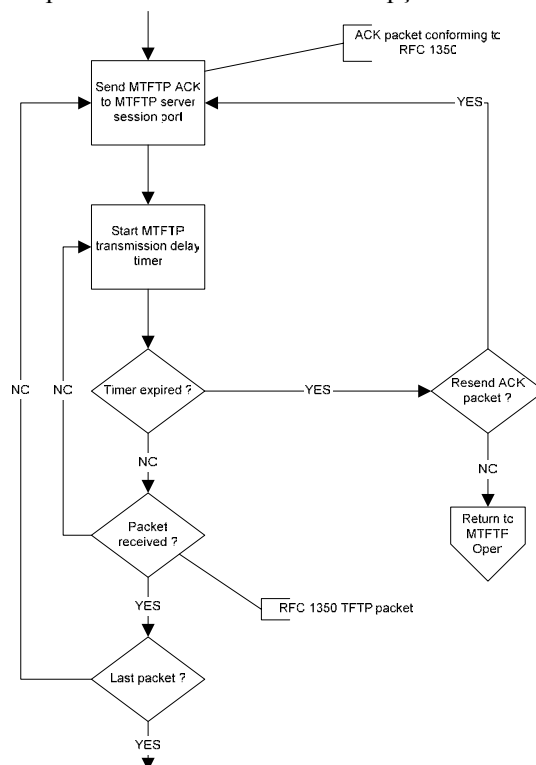



Figura A.7 – Fase de retransmissão/confirmação de pacotes do MTFTP

Este protocolo é um híbrido entre as metodologias de *positive ACK e NACK/ARQ*, porque à medida que recebe os pacotes de dados na 1ª etapa não efectua a confirmação síncrona da sua recepção: o pedido de retransmissão apenas é efectuado quando detectada a falta de um pacote, sendo então precedido de um ACK em bloco de todos os pacotes recebidos. É por este motivo, entre outros, que o uso do protocolo PXE para os sistemas IC^3 está em avaliação: este mecanismo de confirmação intensiva não é recomendável para utilização em ambientes *wireless*. Este facto é ainda agravado pela incapacidade em transportar eficientemente grandes quantidades de informação: acima de um determinado tamanho de ficheiro, a solução passa por aumentar o parâmetro BLKSIZE que, ao exceder o máximo permitido pelo MTU (*Maximum Transfer Unit*) da tecnologia de rede subjacente provoca fragmentação.

No referente ao *Multicast* TFTP, conforme definido pelo RFC2090, aplicam-se as mesmas considerações tecidas para a implementação da Intel.

- o **UDPCast/*Flamethrower*** foi outro dos protocolos analisados. O *UDPCast* [UDPCast] é um sistema de difusão baseado em *reliable multicast*, desenvolvido para replicação de imagens de sistema operativo através de uma rede, para instalações em bloco. Recorre a um protocolo híbrido do tipo NAK com suporte FEC opcional, podendo operar em modo normal, com *feedback* mínimo, ou em modo de *feedback nulo* (chamado modo unidireccional), para ligações unidireccionais ou com latência

elevada. O débito de transmissão é ajustável ao nível do servidor (em função de dois parâmetros: o modo de *duplex* e a taxa de transmissão a utilizar), assim como a agressividade dos códigos FEC, para minimização de *feedback* em função do meio utilizado.

O *UDPCast* segmenta a informação a enviar sob a forma de *slices*, que nada mais são do que uma série de blocos (pacotes UDP). Estes grupos são relevantes para retransmissão de dados (após cada *slice* o servidor questiona os receptores sobre a recepção de todos os blocos e retransmite os que estiverem em falta) e para *Forward Error Correction*. Quando se utilizam códigos FEC (em modo normal ou unidireccional), cada *slice* tem o seu conjunto de blocos de dados e FEC associados. As *slices* são assim divididas em *stripes* (para as quais são calculados e adicionados blocos FEC *interleaved Reed-Solomon* – Figura A.8), encontrando-se organizadas de modo a que blocos consecutivos sejam armazenados em instâncias distintas intercaladas (*interleaving*), diminuindo assim a probabilidade de perdas em bloco.

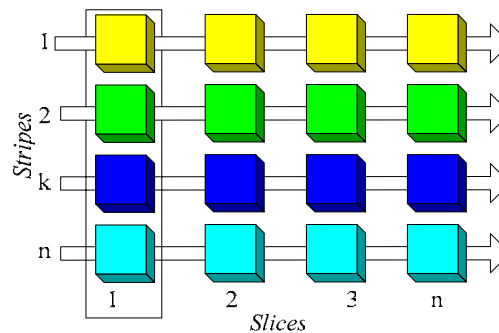



Figura A.8 – *UDPCast*, Segmentação em *Slices* e *Stripes*

O *UDPCast* com suporte FEC pode ser afinado a nível dos seguintes aspectos:

> **Redundância:** influenciando quantos pacotes extra serão enviados por *stripe*. Quanto maior for este valor, maior será a redundância e a robustez da transmissão. Contudo, o *overhead* na comunicação e o tempo de CPU necessário serão também proporcionalmente maiores.
>
> ***Interleave*:** este parâmetro influencia o número de *stripes* a utilizar para a difusão dos dados. Um valor alto aumenta a robustez contra percas em bloco de pacotes, embora de nada sirva quando esta se processa de acordo com padrões aleatórios.
>
> ***Stripe size*:** permite definir quantos blocos de dados estão contidos numa *stripe*. A redução do valor deste parâmetro fornece um modo de aumentar a redundância relativa. Contudo, continua a ser preferível operar com um valor de redundância absoluta superior (maior valor do parâmetro redundância) ao invés de reduzir o *stripe size*, obtendo assim melhor protecção contra perdas agrupadas de pacotes.
> Considerem-se dois casos. Num utiliza-se um mecanismo configurado para modo 4/64 (4 pacotes utilizados para *redundância*, 64 para *stripe size*) oferecendo maior redundância relativa (pela via de um menor *stripe size*). Noutro usa-se o modo 8/128, oferecendo uma maior robustez por via do aumento do número de pacotes para a redundância/FEC. Se, por exemplo, se perderem os primeiros 8 pacotes consecutivos de uma transmissão, então seria impossível recuperar a *stripe* com apenas 4 blocos FEC em 4/64, mas tal seria ainda possível em modo 8/128.

O *flamethrower* [UDPCast] é um *wrapper daemon* escrito em *Perl* que permite manter o componente de transmissão de dados do UDPCast (*udp-sender*) permanentemente activo à escuta de conexões, permitindo que o sistema opere em

push-mode, devendo os receptores estar a postos (ponto de sincronismo) antes de a transmissão ser iniciada.

A operação conjunta do UDPCast/*flamethrower* é ilustrada na Figura A.9.

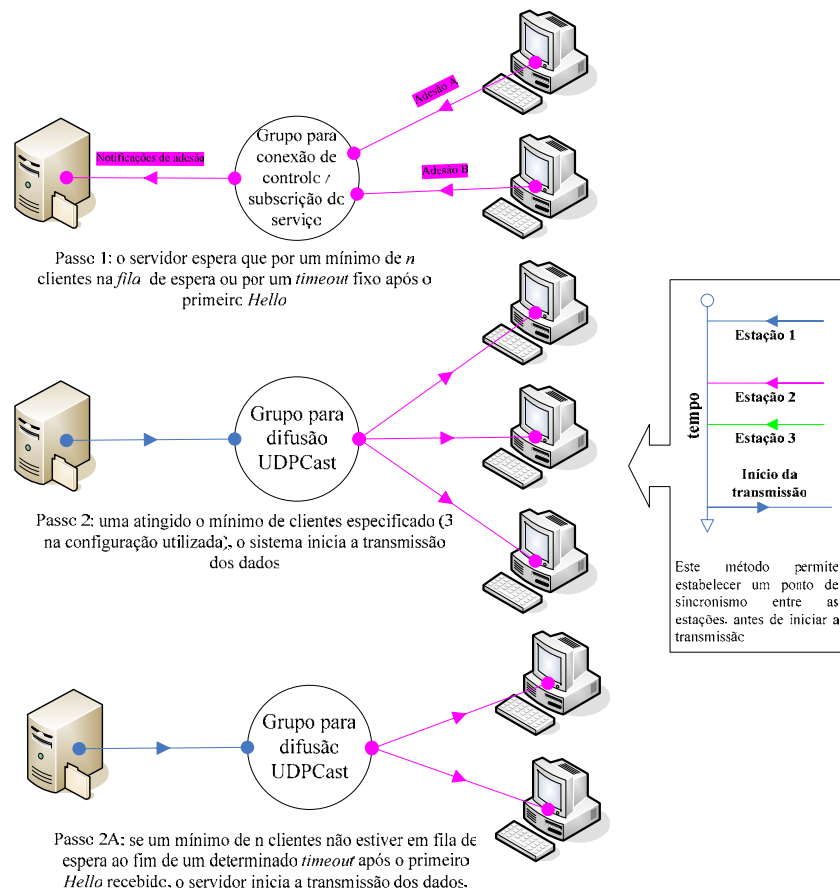



Figura A.9 – Operação Conjunta *UDPCast*/*FlameThrower*

Devido à sua maleabilidade, adequação, estabilidade e maturidade, optou-se no IC^3 por recorrer ao *tandem* UDPCast/*Flamethrower* para difusão *multicast* das imagens de sistema para efeitos actualização. Esta solução afigura-se como a mais adequada para operar de forma pacífica em ambientes *wireless,* visto dispor dos meios necessários para minimizar o tráfego de retorno e lidar com os constrangimentos inerentes ao meio.

B. Suporte VoIP no IC^3

Neste anexo discute-se um conjunto de aspectos relacionados com o suporte VoIP/SIP na plataforma IC^3.

Não pretendendo efectuar um estudo exaustivo sobre o tema, procura-se abordar algumas questões consideradas pertinentes relacionadas com estas tecnologias. Algumas dessas questões estão directamente relacionadas com a plataforma IC^3. Outras extravasam o âmbito especifico da plataforma, e estão mais relacionadas com os serviços VoIP em geral.

Na primeira Secção é discutido o papel do protocolo SIP nas comunicações VoIP. A Secção B.2 identifica alguns dos factores críticos que podem afectar a qualidade de uma infraestrutura VoIP, apontando também potenciais soluções para os problemas inerentes. A terceira Secção discute questões relacionadas com o uso de *Power over Ethernet* para alimentação dos terminais IC^3, tecendo-se também algumas considerações mais genéricas sobre a nova versão do protocolo. Por último, a Secção B.4 identifica os principais riscos e benefícios habitualmente associados à convergência dos serviços de comunicação e computação.

B.1 O Protocolo SIP

O protocolo SIP [RFC3261], utilizado para o suporte VoIP, tem como principais funções o estabelecimento, (re)negociação e término das sessões de comunicação VoIP entre os dois extremos da ligação (podendo estes extremos ser *soft* ou *hard phones*). O SIP opera ao nível da sinalização na camada de aplicação, utilizando a porta 5060 (UDP ou, menos frequentemente, TCP) e não transportando qualquer tráfego de voz. Para o transporte dos fluxos de voz recorre-se ao protocolo RTP operando sobre UDP em portas não-privilegiadas (habitualmente na gama 10000-20000). A relação entre os protocolos SIP e RTP pode ser ilustrada como recurso à chamada "topologia em trapézio", como mostra a figura B.1.

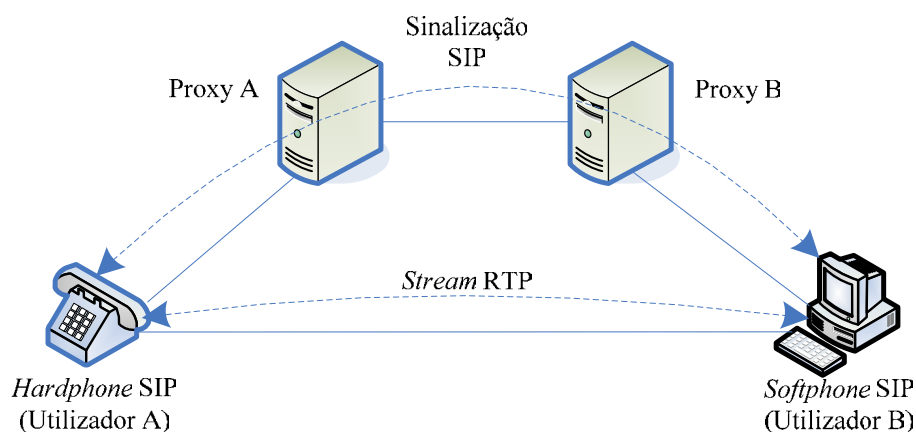



Figura B.1 – Relação entre SIP e RTP

Se o utilizador A pretender fazer uma chamada para B, o *hardfone SIP* contactará o seu proxy SIP (*SoftPBX*) que tentará encontrar B. Neste caso, o utilizador B encontra-se atrás de outro *proxy*, pelo que a negociação da chamada será feita por intermédio dos *proxies* A e B. Uma vez iniciada a chamada, e se tal for possível, a *ligação* de voz será estabelecida de modo a que ambos os extremos comuniquem de forma directa por intermédio de uma *stream* RTP, de modo a economizar os recursos dos *proxies.*

Não obstante o exemplo da Figura B.1, o protocolo SIP também pode operar numa lógica de sinalização directa ponto-a-ponto – com os terminais a comunicar directamente entre si sem intervenção de um *proxy* – ou totalmente indirecta, como ilustra a figura B.2. Neste último caso, existe um *proxied call path* onde a sinalização e tráfego de voz circulam através de *proxies* (no caso do tráfego de voz trata-se de um *back-to-back agent*), nos casos em que é impossível estabelecer uma conexão directa entre os dois extremos da comunicação ou em que é necessário efectuar *transcoding* entre *codecs* distintos.

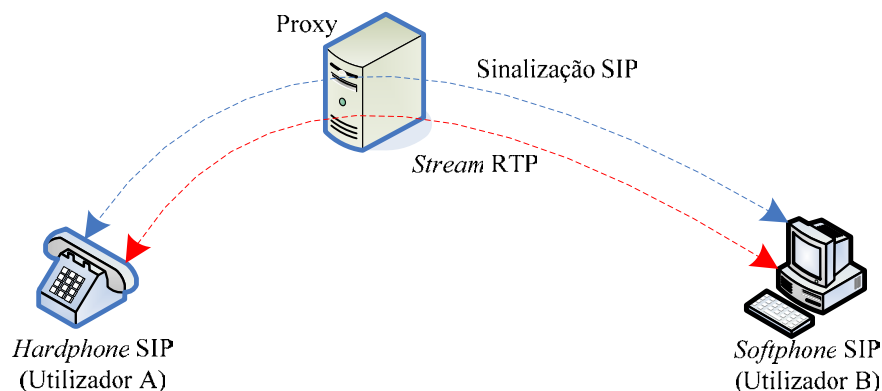



Figura B.2 – Comunicação Totalmente Indirecta

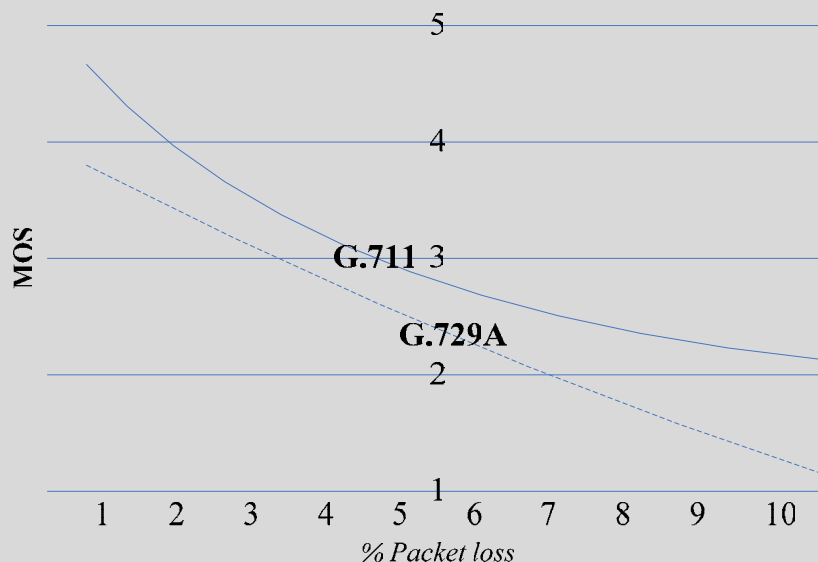
B.2 Qualidade do Serviço VoIP: Precauções e Recomendações

Como foi mencionado na Secção 4.3, o sistema telefónico tradicional utiliza mecanismos de sinalização que evitam, na maioria das situações, que a qualidade das chamadas se degrade em caso de congestão de canal. Contudo, numa rede IP – baseada na tecnologia de comutação de pacotes e não concebida para acomodar comunicações em tempo real orientadas a conexão, como na rede telefónica clássica – a congestão de canal e consequente diminuição de largura de banda disponível podem comprometer de forma crítica a qualidade de uma chamada VoIP em progresso.

Assim, existe a clara necessidade de planeamento e dimensionamento antes de implementar uma solução VoIP a nível organizacional, utilizando para o efeito um conjunto de métricas adequado (*cfr. caixa*).

Avaliação da Qualidade das Comunicações VoIP

Por tradição, a qualidade das ligações telefónicas é medida de acordo com a escala MOS (*Mean Opinion Score*), sendo o valor obtido (na escala de 1 a 5) calculado a partir da opinião de um grupo (amostra) de utilizadores do serviço, ao qual é dado a ouvir um conjunto de extractos sonoros obtidos a partir de chamadas de qualidade variável, registadas sob as mais diversas condições. No respeitante ao VoIP, as condicionantes a considerar para aferir/classificar as variações de qualidade das amostras sonoras são normalmente dependentes dos *codecs*, dos procedimentos de *transcoding* ("tradução" entre *codecs* distintos), do *packet loss* e da latência.

Apesar de não ser necessário na maioria dos casos recorrer à execução de testes MOS, devem ser levados em conta os resultados obtidos em testes conhecidos (e confiáveis) no planeamento da solução VoIP a implementar. A figura que se segue foi obtida a partir de um estudo da *Nortel Networks* (citado em [Wallingford 2005]) e relaciona o *packet loss* com a qualidade perceptível das chamadas, de acordo com o *codec* utilizado.

A interpretação deste gráfico permite obter um conjunto de informações de grande relevância:

- Apesar das vantagens que lhe são reconhecidas (codificação de voz com reduzido *bitrate*), o *codec* G.729 (comercial, com esquema de licenciamento bem definido) não se apresenta como uma alternativa viável para implementação numa LAN, uma vez que a qualidade perceptível pelos utilizadores não é de forma alguma compatível com o que se espera de uma infraestrutura local e controlada integralmente pela organização.
- O *codec* G.711 (gratuito) oferece melhor qualidade de comunicação em redes LAN, onde os benefícios tradicionais do G.729 não são relevantes. Contudo, o gráfico mostra que apesar da necessidade de um *bit rate* mais elevado não ser obstáculo para o *codec* G.711 numa rede de área local, o *packet loss* continua a ser um factor crítico para a qualidade percepcionada.

Ainda que na maioria das vezes não sejam necessários estudos MOS, considera-se que mesmo para infraestruturas de média dimensão (dezenas de utilizadores) se deve aferir a qualidade do serviço prestado por meio de inquéritos aos utilizadores, antes da implantação da solução VoIP – para ter um ponto de comparação – e depois, com regularidade, para aferir a qualidade do serviço prestado. Após a implantação da solução VoIP, e dependendo da dimensão da infraestrutura, poderá mesmo fazer sentido utilizar a métrica MOS para estabelecer *Service Level Agreements* com os clientes do serviço VoIP, definindo uma expectativa para cada *call path* baseada na escala MOS.

Outra situação a evitar será o excessivo sobredimensionamento de capacidade da infraestrutura de comunicações em nome da melhoria da qualidade de serviço prestado. Esta técnica pode resultar para tráfego tradicional (por exemplo tráfego *web* ou *queries* a bases de dados) mas é inadequada para um serviço VoIP, em consequência da natureza da sua operação em tempo (quase) real, num fluxo isócrono.

B.2.1 Factores Críticos Relacionados com a Infraestrutura de Rede

Para o VoIP, e falando em termos da infraestrutura de rede, existe um conjunto bem definido de factores críticos que afectam de forma determinante a qualidade das comunicações.

Um desses factores é a **latência**. Em VoIP a latência extremo a extremo (o tempo decorrido entre a emissão de um som pelo emissor e a sua recepção) é considerada um factor crítico. De acordo com a recomendação ITU-T G.114 o valor máximo aceitável para latências *end-to-end* é de 150ms, embora alguns fabricantes considerem tolerável a faixa entre os 150ms e os 300ms (se bem que desaconselhada – fabricantes como a *Cisco Systems* consideram valores acima dos 150ms como totalmente inaceitáveis, com base em estudos que demonstram que latências RTT acima dos 250ms são perceptíveis pela maioria dos utilizadores) e totalmente inaceitável acima deste intervalo. Valores exagerados de latência produzem diversos efeitos nefastos:

- Abrandamento do ritmo da conversação.
- Interrupção involuntária entre ambos os intervenientes na chamada.
- Agravamento do eco.
- Problemas de sincronização em conferências com vários utilizadores.

A latência pode também ser agravada como resultado da aplicação de algoritmos de *buffering* (para eliminar o *jitter*) e recuperação de *packet loss*. De facto, a latência pode derivar de:

- Processamento para transmissão por rede (*packetization* e *framing*, processo de *marshalling* que pode consumir até 30ms).
- Limitações físicas, por exemplo ao nível do *interface* entre a camada de rede lógica e a rede física. O acto de serializar os dados por forma a serem colocados no meio físico de comunicação implica um atraso que é inversamente proporcional à velocidade da ligação (quanto mais rápido o meio, menor a latência). Este atraso é incontornável e pode ser minimizado pela redução do número de *links* intervenientes na comunicação e pelo uso de ligações com largura de banda adequada (*cfr caixa*).
- Propagação. O atraso de propagação dos dados serializados no meio físico deve ser considerado quando estes tiverem de percorrer uma grande distância.

Relacionando o débito e a latência das comunicações

Por vezes as diferenças no respeitante ao *packet delay* entre duas linhas de acesso com débito distinto não são óbvias. Exemplificando para tráfego VoIP sobre linhas de 56Kbps, E1 e *Fast Ethernet* (*codec* G.711 com 1 *sample* por pacote, *stream* RTP sobre UDP com 280 *bytes* por pacote, *overhead* incluído), os valores são os seguintes:

Linha de 56Kbps	**Linha E1** (2Mbps divididos em 32 canais de 64Kbps, dos quais 2 – 128Kbps - estão reservados para sinalização e controlo)	**Fast Ethernet 100Mbps** (42 bytes de *overhead* por pacote: 12 *gap* + 8 preâmbulo + 18 *header* (c/802.1q) + 4 *trailer*)
(280 *octetos* * 8 *bits*)/56Kbps = 40ms *ingress + egress delay* = 40 * 2 = 80ms	(280 * 8)/1920Kbps = 1,17ms *ingress + egress* = 2,34ms	((280 + 42) * 8)/100Mbps = 0,026ms *ingress + egress* = 0,052ms

- *Queuing*. O tempo de *buffering* a que cada pacote é submetido antes de ser transmitido é também fonte de latência. Este atraso depende das condições de carga da rede (se houver congestão de rede os pacotes estarão mais tempo na fila, à espera de ser transmitidos) e, quando o *buffer* de *queueing* for configurável, deverá ser ajustado de modo a não crescer excessivamente, provocando problemas de latência.
- Processamento por *software* (filtros para remoção e ruído e eliminação de eco, por exemplo), *codecs* e tratamento de perdas.
- B*uffering* para normalização de *jitter*.
- T*ranscoding*. Quando os dois extremos da comunicação operam com *codecs* distintos o *SoftPBX* efectua a "tradução" de um para o outro e vice-versa. A Tabela B.1 mostra os tempos medidos, em *ms*, para a transposição entre um grupo de *codecs* geralmente utilizados, usando para o efeito um *SoftPBX Asterisk*.
- R*outing* (5 a 50ms por nó intermédio, em média)*, packet-switching* (a latência é maior no caso do *buffering* em *switches store-and-forward*, por oposição ao método *cut-through*, embora neste último tipo não seja efectuado o cálculo do CRC e verificação de erros) e transposição de *firewall* (o processamento de regras para tratamento de pacotes num *firewall* pode ser uma importante fonte de atraso).

Codec	**GSM**	**G.711 ULaw**	**G.711 ALaw**	**G.726**	**LPC10**	**ILBC**
GSM	N/A	4	4	12	14	59
G.711/ULaw	10	N/A	1	10	12	57
G.711/ALaw	10	1	N/A	10	12	57
G.726	17	2	2	N/A	19	64
LPC10	18	10	10	18	N/A	65
ILBC	19	11	11	19	21	N/A

Latências em ms, medidas num servidor Asterisk com kernel Linux 2.4.20-6, CPU Pentium III a 600MHz[Wallingford 2005]

Tabela B.1 – Latências Medidas na Transposição Entre *Codecs*

A **perda de pacotes** é também um importante factor para a qualidade das comunicações VoIP. A tolerância dos *codecs* a este problema é variável, chegando mesmo alguns a embeber mecanismos de *concealing* (PLC – *Packet Loss Concealement*) que conjugam técnicas de interpolação matemática com modelos psico-acústicos para tentar "preencher" o som em falta num dado pacote perdido, usando para o efeito o resultado da análise dos pacotes recebidos antes e depois da perda. No entanto, mesmo o recurso a *codecs* com suporte PLC não evita que numa rede VoIP a taxa de pacotes perdidos deva ser mantida abaixo de 1%. O recurso a mecanismos de QoS (*Quality of Service*) e/ou CoS (*Class of Service*), com o objectivo de reservar capacidade de rede e/ou dar prioridade ao tráfego VoIP, ajuda a solucionar esta questão. São igualmente efectivas medidas para conservar a capacidade da rede e limitar as perdas a um mínimo.

O ***jitter*** é outro facto relevante. O seu efeito pode ser atenuado com recurso a técnicas de QoS/CoS e a dispositivos designados por *jitter buffers*, colocados em pontos extremos e nos servidores VoIP. No entanto, esses dispositivos penalizam a latência.

B.2.2 Factores Críticos Adicionais

Além dos factores críticos relacionados com a rede, existem outros que podem ter causas alheias (ou não) à infraestrutura de comunicações, nomeadamente:

- **Ruído**. Numa comunicação VoIP o ruído pode ter várias proveniências, desde as de origem física (ruído ambiente, interferência eléctrica ou estática) às provocadas pelo tratamento por filtros e/ou *codecs* (sobremodulação, distorção por amplificação, ou no

próprio processo de quantização[9]/*sampling*) passando pelas causas derivadas das condições da rede (latência, *jitter*, perda de pacotes). Para cada causa existe um conjunto de soluções específicas que podem ser postas em prática, que vão do recurso a filtros de eliminação de ruído (por *software* e/ou *hardware*) até ao uso de métodos de QoS/CoS.

- **Eco.** De forma análoga ao ruído, o eco pode ter origem origem física (proximidade de microfone e auscultador, eco gerado na *interface* analógica/TDM-*SoftPBX* e vice--versa) ou ser derivado de problemas relacionados com a rede de comunicações de dados (por exemplo latência RTT entre dois extremos de uma comunicação). Existem algoritmos de supressão de eco bastante eficazes que podem ser implementados a nível do SoftPBX e dos clientes SIP (*softphones*), encontrando-se também alguns auriculares e microfones equipados com mecanismos mais ou menos rudimentares de eliminação de *feedback*.

B.2.3 Mecanismos de Rede Relevantes para o Serviço VoIP

Para controlar e conter os efeitos nefastos identificados nas Subsecções anteriores, é possível recorrer a diversos mecanismos.

A nível de CoS podem ser implementados e configurados mecanismos de prioritização de tráfego, tais como:

- **802.1p/ToS (*Type Of Sevice*).** A norma IEEE 802.1p [Lidinsky 1999] baseia-se no uso de um subcampo de 3-bits dos pacotes *ethernet* (nomeadamente do campo *ToS*) para classificar cada pacote, de modo a atribuir níveis de precedência distintos. Normalmente, o suporte 802.1p é incluído a nível dos *switches* de camada 2, permitindo classificar até 8 categorias distintas de tráfego (o máximo possível com 3 bit). O comportamento *per-hop* associado a cada classe é definido pelo gestor da rede. Associa-se frequentemente a Classe 5 ao tráfego VoIP.

- ***Diffserv (Differentiated Services).*** Operando a nível do encaminhamento ponto-a--ponto entre redes WAN (e portanto, preferencialmente em *edge routers*), o *diffserv* [RFC2475] utiliza o campo *ToS* para marcar e identificar a política de encaminhamento e prioritização a adoptar para os pacotes durante a circulação destes no interior da rede. Uma vez admitido um pacote ao interior da rede, todos os outros *routers* deverão respeitar e fazer cumprir a política de precedência *diffserv* marcada no campo *ToS* pelo *edge router* que admitiu o pacote na rede. As classes são normalmente codificadas em 3 bits (embora esteja previsto o uso de 6 bits, nalgumas situações), recorrendo assim ao subcampo de precedência do *ToS* para o efeito. As classes definidas podem ser associadas em três grupos essenciais (chamados de classes DSCP, PHB ou de tráfego): AF/*Assured Forwarding*, EF/*Expedited Forwarding*, e BE/*Best Effort*.

  É possível criar um "ponto de decisão" *diffserv* no próprio servidor do *SoftPBX Asterisk*, utilizando para o efeito a *framework netfilter* de manipulação e filtragem de pacotes IP existente no *kernel Linux*, classificando (e prioritizando) assim o tráfico VoIP que entra e sai do servidor de comunicações. O seguinte comando, por exemplo, permite associar ao tráfego RTP de entrada e saída (porta 5004) à classe *diffserv* EF.

```
iptables –A PREROUTING –p udp –D 0.0.0.0/0.0.0.0
              –dport 5004 –j DSCP \ –set-dscp-class EF
```

A nível de mecanismos de QoS, as soluções baseadas em protocolos como o RSVP/*Intserv* ou mesmo em tecnologia MPLS podem ser aplicadas em organizações com múltiplas filiais, geograficamente dispersas e com elevados níveis de exigência em termos de qualidade e fiabilidade. Contudo, para o cenário considerado (rede local), não se preconizam outras

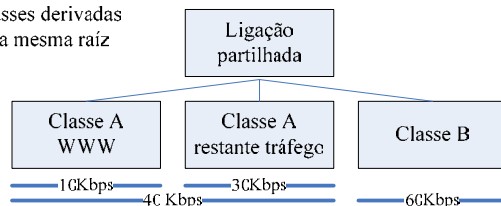
[9] Processo que permite passar do domínio contínuo para o discreto, podendo ser definido como a interpretação de uma quantidade contínua com recurso a um conjunto finito de valores discretos.

medidas além das baseadas em CoS, pela sua simplicidade e eficácia. Numa rede local na qual cerca de 30% do tráfego gerado seja de voz [Wallingford 2005], por exemplo, é habitualmente suficiente usar ToS/802.1p, suportado pela maioria dos *switches ethernet*.

B.2.4 Mecanismos de QoS ao Nível do Terminal

No entanto, mesmo aplicando estes mecanismos ao nível da rede, continua a ser possível que um determinado posto de trabalho sofra de escassez de recursos de rede devido a tráfego demasiado intenso, seja esse tráfego intencional ou involuntário. Nestas situações seria interessante reservar permanente alguma capacidade da rede para tráfego VoIP/H.323, de modo a assegurar a disponibilidade e/ou resiliência desta categoria de serviços SoIP. Foi também por este motivo que se enunciou o requisito funcional do suporte para QoS/Cos na Secção 4.3, tendo em vista a possibilidade de tratar a questão da qualidade do serviço tanto a nível global como no próprio posto de trabalho.

O *kernel* do sistema operativo *Linux* – usado nos terminais protótipo do IC³ – incorpora diversos mecanismos de controlo de tráfego (*traffic policing* e *shaping*), dos quais se destaca o HTB [HTB] como o mais adequado para o propósito pretendido[10].

A disciplina de *queuing* HTB baseia-se na divisão do tráfego em classes, sendo atribuída a cada classe uma fatia de largura de banda (parâmetro *rate*, análogo ao *Commited Information Rate* do *Frame Relay*) e um tecto máximo (parâmetro *ceil*). As classes estão organizadas de forma hierárquica, sendo o tráfego em excesso entre o *rate* e o *ceil* subtraído ao total atribuído à classe imediatamente acima e, portanto, ao excedente das restantes classes dependentes da mesma raiz.

B.2.5 Mecanismos de QoS ao Nível do Terminal, HTB

Para melhor compreender o HTB atente-se nos exemplos ilustrados na Figura B.3, considerando um *router* baseado em *Linux*, que recorre a HTB para fazer *traffic shaping* na partilha de uma ligação entre o sistema A e o sistema B.

Na **Hipótese 1** ilustrada na Figura B.3, o *router* foi configurado para conceder 40Kbps para o sistema A e 60Kbps para o sistema B, sendo os 40Kbps do sistema A segmentados em 30Kbps para tráfego *Web* e 10Kbps para as restantes aplicações. São criadas as classes HTB, com os seguintes comandos:

```
tc qdisc add dev eth0 root handle 1: htb default 12
tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 30kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:11 htb rate 10kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:12 htb rate 60kbps ceil 100kbps
```

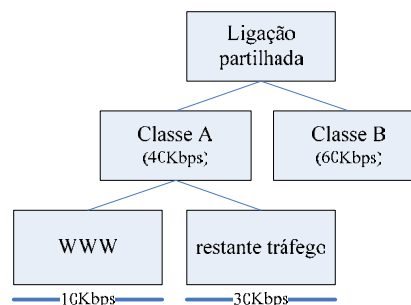



Figura B.3 – Exemplos de Uso de HTB

[10] Além do HTB é de mencionar o CBQ [Floyd 1995], um mecanismo semelhante e também bastante disseminado, que é no entanto mais complexo de configurar sem acarretar benefícios adicionais.

É em primeiro lugar criado um *handle HTB* para o interface de rede *eth0* (*handle* "*:1*"**)**. É também definido que todo o tráfego que não seja alvo de classificação será associado à classe "*1:12*". É depois criada uma classe "*1:1*" (raiz, associada ao *handle* "*:1*") com um *rate* de 100Kbps e um *ceil* de 100kbps. São depois criadas três subclasses de subclasses de "*1:1*": "*1:10*" (A/WWW), "*1:11*" (A/restante tráfego) e "*1:12*" (B). Esas subclasses apresentam *rate* de 30Kbps, 10Kbps e 60Kbps, respectivamente, e *ceil* de 100Kbps.

Podem de seguida ser executados os seguintes comandos:

```
tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip src\ 1.2.3.4
match ip dport 80 0xffff flowid 1:10
tc filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip src\ 1.2.3.4
flowid 1:11
```

O primeiro commando associa a classe "*1:10*" ao tráfego "http/porta 80" do Sistema A (endereço IP 1.2.3.4). O segundo associa a classe "*1:11*" ao restante tráfego proveniente do mesmo sistema, para o qual estão reservados 10Kbps.

Para o Sistema B não há necessidade de mais comandos, pois todo o tráfego não classificado será atribuído à classe "*1:12*" e, portanto, ao Sistema B**.**

Nesta hipótese qualquer excesso de tráfego – em relação ao *rate* associado a cada classe – pode "usar" tráfego não utilizado por outras.

Na **Hipótese 2**, recorre-se a uma hierarquia de classes definida do seguinte modo:

```
tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:2 htb rate 40kbps ceil 100kbps
tc class add dev eth0 parent 1:2 classid 1:10 htb rate 30kbps ceil 100kbps
tc class add dev eth0 parent 1:2 classid 1:11 htb rate 10kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:12 htb rate 60kbps ceil 100kbps
```

Criam-se assim duas classes derivadas da raíz ("*1:2*" e "*1:12***"**, correspondente ao Sistema B), com valores de *rate* de 40 e 60Kbps, respectivamente. A classe "*1:2*" é ainda dividida em duas subclasses: "*1:10*" (A/WWW) e "*1:11*" (A/restante tráfego), com 30Kbps e 10Kbps, respectivamente. Associam-se posteriormente as subclasses "*1:10*" e "*1:11*" a tráfego *http* e a ao restante tráfego (*slack*) gerado pelo Sistema A, com comandos análogos aos da Hipótese 1. Destaque-se a aplicação de uma regra implícita que dita que a soma dos valores de *rate* de duas subclasses deve ser igual ao *rate* de tráfego atribuído à classe mãe. Se A e B fossem clientes distintos esta solução garantiria que B nunca poderia por exemplo aproveitar uma folga ao nível do tráfego da subclasse A/WWW (que pode no entanto ser aproveitado para o *slack* do mesmo cliente, não excedendo o total de 40Kbps total atribuído ao Sistema A).

Como se depreende, esta configuração ainda permite que o Sistema B possa "recuperar" parte da largura de banda não utilizada pelo Sistema A, até ao limite da conexão (100Kbps). A solução passa por algo semelhante aos seguintes comandos:

```
tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbps ceil 100kbps
tc class add dev eth0 parent 1:1 classid 1:2 htb rate 40kbps ceil 60kbps
tc class add dev eth0 parent 1:2 classid 1:10 htb rate 30kbps ceil 60kbps
tc class add dev eth0 parent 1:2 classid 1:11 htb rate 10kbps ceil 60kbps
tc class add dev eth0 parent 1:1 classid 1:12 htb rate 60kbps ceil 70kbps
```

Com esta configuração o Sistema A pode utilizar toda a largura de banda não usada pelo Sistema B, até um total de 60Kbps (excedendo assim os 40Kbps do rate máximo atribuído) e B pode fazer o mesmo, até um máximo de 70Kbps.

B.2.6 HTB Complementado por *Shapers*

Por si só, o uso de HTB poderia responder ao problema da reserva de largura de banda para o VoIP. Contudo, existe ainda uma outra solução que é efectivamente utilizada no sistema

desenvolvido, em conjunto com este mecanismo: filas de prioridade. O HTB, associado ao sistema de prioridades do *shaper* do *kernel linux* (que permite, entre outras coisas, prioritizar a atribuição da largura de banda residual de outras classes, conforme o desejado) permite por exemplo criar uma configuração como a que ilustra a Figura B.4.

No cliente – estratégias idênticas podem ser aplicadas ao próprio SoftPBX – é privilegiado o tráfego VoIP, concedendo-lhe não só uma reserva de largura de banda (o *slack* da classe HTB associada à prioridade 2, cujo resíduo não utilizado é atribuído de forma prioritária para o tráfego VoIP) como um grau de precedência que lhe confere prioridade sobre os outros tipos de tráfego. Quando um esquema de prioridade como o esquematizado se encontra em uso são criadas filas para recepção dos pacotes, para cada classe estabelecida. Quando os pacotes são removidos da fila o sistema só retirará um elemento da fila da classe 2 se não existir nenhum na classe 1. A adopção desta técnica permite também atenuar a latência, tão importante no contexto de VoIP.
Observe-se que a parametrização deste sistema de prioritização é feita pelo administrador, não podendo o utilizador aceder à sua configuração. Isto permite impor políticas globais de gestão de tráfego numa infraestrutura organizacional que, ao contrário do normal, começam logo a ser aplicadas ao nível dos pontos terminais. Isto abre as portas à implementação de mecanismos distribuídos de controlo automático dos níveis de utilização da rede, que podem intervir ao nível dos próprios postos de trabalho para evitar e controlar situações de congestão de tráfego.

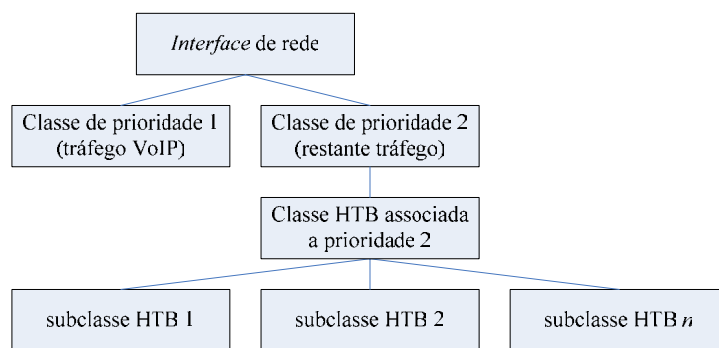



Figura B.4 – Exemplo de Configuração de HTB Complementada por *Shapers*

B.2.7 Escolha de *Codecs*

Para terminar esta subsecção, refira-se a questão dos *codecs* e da sua importância.

Como foi já mencionado, o *codec* G.711 aparenta um bom grau de resiliência associado à qualidade de som, pontuando na escala MOS com valores entre 4,1-4,4 a 2 aproximadamente). Assim, e face a outros *codecs* testados, optou-se por usar G.711 na plataforma IC^3, dadas as suas características: boa qualidade de som numa *stream* de aproximadamente 64Kbps, codificada em PCM monaural 8bit *Alaw* ou *Ulaw*; sem compressão; pouco peso em termos de CPU; adequação para uso em redes locais e situações de interoperabilidade com a estrutura telefónica tradicional.

Por motivos de retrocompatibilidade e suporte para ligações de reduzido débito, incluiu-se também o GSM: qualidade de som razoável, monaural, codificado num fluxo de aproximadamente 13Kbps, pouco intensivo em termos de uso de CPU quando comparado com outros algoritmos da mesma categoria, como o G.729.

A escolha destes dois *codecs* teve também em conta o custo do *transcoding* entre eles, que é dos mais reduzidos. Especial atenção deve também ser dada ao facto da eficiência dos *codecs* utilizados poder variar de acordo com um determinado número de parâmetros a ter em conta, como atestam as tabelas B.2 e B.3.

***Codec*: G.711, tamanho por *sample* de 240 *bytes*, duração por *sample* de 30ms**

***samples*/ pacote**	**pacotes/ seg**	**Tam. *payload* Bytes**	**bits/ seg**	**tam. pacotes bytes**	**bits/ seg**	**% Overhead**	**latência ms**
1	33,33	240	64000	280	74667	17%	30,00
2	16,67	480	64000	520	69333	8%	60,00
3	11,11	720	64000	760	67556	6%	90,00
4	8,33	960	64000	1000	66667	4%	120,00
5	6,67	1200	64000	1240	66133	3%	150,00
6	5,56	1440	64000	1480	65778	3%	180,00
7	4,76	1680	64000	1720	65524	2%	210,00
8	4,17	1920	64000	1960	65333	2%	240,00
9	3,70	2160	64000	2200	65185	2%	270,00
10	3,33	2400	64000	2440	65067	2%	300,00
11	3,03	2640	64000	2680	64970	2%	330,00
12	2,78	2880	64000	2920	64889	1%	360,00

Tabela B.2 – Eficiência de *Codecs*; G.711 (adaptado de [OpenH323])

Uma análise a estas tabelas demonstra a necessidade de não menosprezar a parametrização do uso dos *codecs*. Factores críticos como o *overhead*[11] e *payload* efectiva por pacote devem ser considerados tendo em vista a obtenção de níveis adequados de eficiência no uso do *codec* seleccionado, operando dentro de limites de latência aceitáveis com consumo óptimo de largura de banda (o aumento da *payload* de voz por pacote permite obter uma redução no consumo de largura de banda à custa do aumento da latência e da diminuição da resiliência, pois se a perda de um pacote transportando 20ms de som numa conversação é practicamente imperceptítel tal já não será verdade para uma *payload* de 60ms). Para o *codec* G.711 numa rede sem problemas de congestionamento de tráfego e com largura de banda disponível será por exemplo recomendável operar com *samples* de 20-30ms a 1-2 *samples* por pacote, resultando em latências da ordem dos 20-60ms e ocupação de largura de banda efectiva que pode chegar, no caso extremo aos 74/80Kbps (para *samples* de 30/20ms, respectivamente). Se a estes cálculos se acrescentar o *overhead* da tecnologia de transporte da camada física os valores tenderão para níveis ainda mais elevados. A Figura B.5 ilustra esta situação para uma rede local baseada em tecnologia Ethernet.

***Codec*: GSM, tamanho por *sample* de 33 *bytes*, duração por *sample* de 20ms**

***samples*/ pacote**	**pacotes/ seg**	**tam. *payload* bytes**	**bits/ seg**	**tam. pacotes bytes**	**bits/ seg**	**% overhead**	**Latência ms**
1	50,00	33	13200	73	29200	121%	20,00
2	25,00	66	13200	106	21200	61%	40,00
3	16,67	99	13200	139	18533	40%	60,00
4	12,50	132	13200	172	17200	30%	80,00
5	10,00	165	13200	205	16400	24%	100,00
6	8,33	198	13200	238	15867	20%	120,00
7	7,14	231	13200	271	15486	17%	140,00
8	6,25	264	13200	304	15200	15%	160,00
9	5,56	297	13200	337	14978	13%	180,00
10	5,00	330	13200	370	14800	12%	200,00
11	4,55	363	13200	403	14655	11%	220,00
12	4,17	396	13200	436	14533	10%	240,00

Tabela B.3 – Eficiência de *Codecs*; GSM(adaptado de [OpenH323])

[11] O protocolo SIP só estabelece o método de sinalização, sendo a *stream* de som transportada através de pacotes RTP sobre UDP. Assim sendo, acresce a cada pacote 20 bytes de cabeçalho IPv4 + 8 bytes de *cabeçalho* UDP + 12 bytes por *sample* = 40 bytes (320 bits) de *overhead.* Não se considerou o *overhead* da tecnologia de rede utilizada para transporte nos cálculos das tabelas.

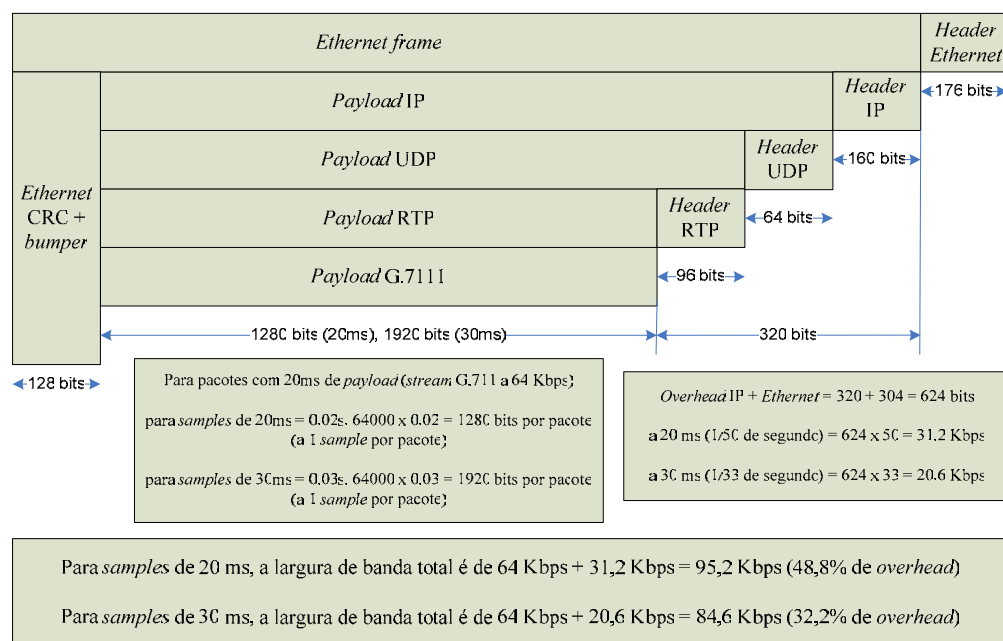



Figura B.5 – Eficiência do VoIP em redes *Ethernet*

B.3 Normas *Power over Ethernet* 802.3af e PoEP

Um terminal telefónico tradicional recebe alimentação eléctrica a partir da infraestrutura da própria cablagem de voz. Em caso de corte de corrente, o PBX – presumivelmente equipado com baterias de *backup* – mantém as extensões operacionais por um período de tempo que, tipicamente, vai de 2 a 8 horas. Os utilizadores estão habituados a esta disponibilidade e contam com ela, por exemplo, para fazer chamadas de emergência em situações de corte de corrente.

Nesta perspectiva, e no contexto da Plataforma IC³, seria extremamente interessante ter terminais fixos passíveis de alimentação a partir da própria tomada de rede *ethernet*, recorrendo para o efeito, por exemplo, à norma 802.3af (*Power over Ethernet,* ou PoE [IEEE 2003]). No entanto, como se verá, ainda não é viável, com o estado actual da tecnologia, dispor de terminais convergentes (posto de trabalho e terminal de voz) passíveis de alimentação por PoE.

A arquitectura estabelecida pela norma 802.3af estrutura-se em torno de dois conceitos básicos: a fonte de energia e o equipamento alimentado. Considera-se equipamento alimentado (*Powered Device*) o dispositivo que requer alimentação por meio da ligação *ethernet*, podendo a potência fornecida atingir os 12,95 Watt, de acordo com a norma actual.

Considera-se como fonte de energia (PSE – *Power Sourcing Equipment*) o equipamento que fornece energia – efectuando a pesquisa para detecção de dispositivos que dela necessitem, monitorizando a linha e fornecendo alimentação se e apenas quando um terminal conforme a norma 802.3af for detectado no outro extremo. O PSE fornece até 15,40 Watt a uma tensão de 44-57 V DC por terminal (a discrepância entre este valor e a potência efectivamente disponível no equipamento alimentado justificam-se pelas perdas de transmissão). A função de PSE pode ser efectuada por um equipamento activo de rede conforme com a norma 802.3af (um *switch ethernet,* por exemplo) ou através de um "injector" específico.

Com apenas 13 Watt disponíveis por porta *ethernet*, e com os consumos típicos de um posto de trabalho, é ainda impraticável alimentar por PoE os terminais IC³. De acordo com a estimativa apresentada na Secção 4.2.1, o consumo do protótipo IC³ – que apresenta já uma considerável eficiência, quando comparado com o PC tradicional – ronda os 33,20 Watt, não incluindo o monitor.

A eventual solução poderia assim passar por uma de três alternativas:

- **Reduzir o consumo de energia do terminal**. Confirme foi já discutido na Secção 4.2.1, um protótipo construído em torno da *motherboard* EPIA 5000 e leitor/gravador de CD poderia manter os requisitos energéticos abaixo dos 13W. A viabilidade desta alternativa é no entanto condicionada de forma decisiva pelas limitações de processamento (processador C3 533MHz) e pela necessidade que continua a existir de alimentar o monitor.

- **Aguardar pela norma *802.3at (Power over Ethernet Plus)*.** o grupo 802.3 do IEEE encontra-se presentemente a trabalhar na elaboração de uma nova geração da tecnologia PoE, denominada 802.3at (*Power Over Ethernet Plus* [Di Minico 2005]), capaz de suportar cargas superiores a 30 Watt por porta. Com esta tecnologia seria possível alimentar o *s*istema *IC³* através da tomada de rede.

  Contudo, esta norma ainda está em definição e não é garantido que a promessa de 30 Watt por porta seja efectivamente cumprida, devido às limitações inerentes à cablagem normalmente utilizada na infraestrutura de rede (*cfr. caixa*).

- **Acoplar uma bateria de pequena capacidade à PSU externa.** Existe ainda uma possibilidade, baseada no recurso à técnica do *trickle charging*[12], que permitiria conceber uma fonte de alimentação equipada uma bateria de pequena capacidade conectada ao sistema em permanência e capaz de sustentar a operação do sistema por alguns minutos, em caso de corte de corrente. Há por exemplo registo de uma experiência onde se manteve um terminal EPIA5000 em funcionamento – disco rígido incluído – durante 138 minutos com 24 pilhas alcalinas tipo D [LinITX]. Ainda que fosse obviamente possível usar uma unidade de alimentação ininterrupta clássica (UPS) por terminal, a solução integrada seria significativamente mais simples e mais ergonómica.

Em termos de viabilidade, consideram-se as duas últimas alternativas como as mais capazes, embora nenhuma delas possa ser imediatamente executada, quer por limitações de tempo (para a adição da bateria à PSU), quer por indisponibilidade (o *standard* 802.3at não está ainda definido).

Isto significa que a disponibilidade em caso de emergência é um problema ainda por resolver. Para infraestruturas de maiores dimensões este problema é suficientemente vasto para justificar um estudo *per se*, que ultrapassa o âmbito deste documento.

Entretanto, para pequenas infraestruturas nas quais não seja viável cobrir todo o equipamento por alimentação ininterrupta, poderá optar-se por soluções de compromisso, tais como a manutenção, em paralelo com a infraestrutura VoIP, de linhas de *backup* tradicionais directamente ligadas ao exterior ou conectadas ao *interface* do SoftPBX com a estrutura telefónica analógica/TDM através de um *pass-through* que funcione mesmo se o SoftPBX estiver desligado.

[12] Técnica de carga em que o carregador fornece uma corrente constante (mas de débito muito baixo – da ordem do centésimo da capacidade da bateria) à bateria, independentemente do estado ou temperatura desta para compensar as perdas de energia e descargas por uso intermitente. Esta técnica pode ser implementada recorrendo a um circuito muito simples, complementado com mecanismos de limitação de carga por monitorização do estado da bateria, para evitar degradação das características desta.

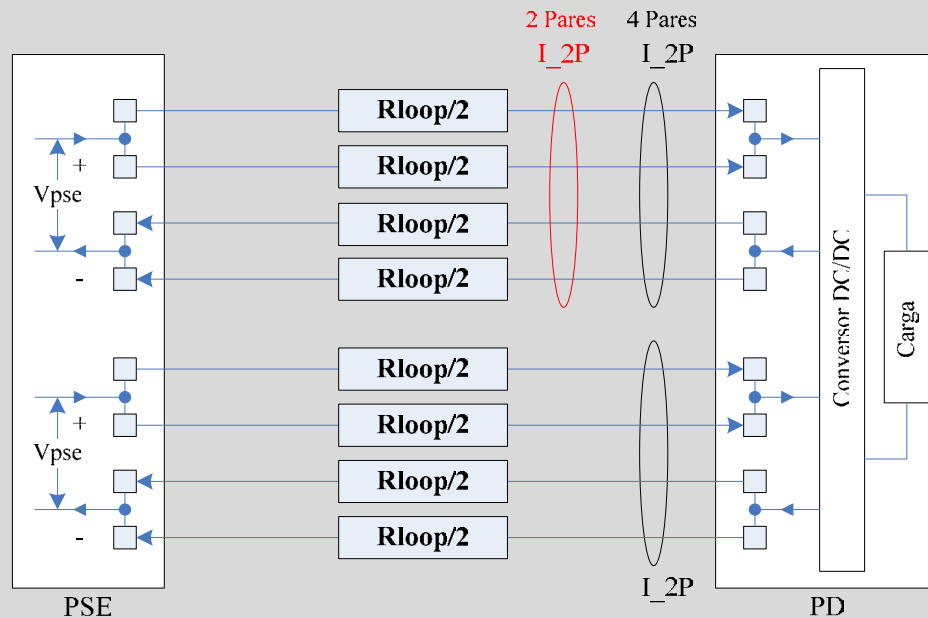
A Física: Inimiga de Uma Boa Ideia?

Os principais problemas com que a *task force* da norma IEEE 802.3at se debate prendem-se com as limitações da própria cablagem UTP CAT5. A *ampacity* (capacidade de transporte de corrente eléctrica) é normalmente definida na indústria das cablagens em função dos seguintes aspectos:

- Calor gerado no condutor.
- Calor gerado no isolamento.
- Calor gerado na malha de isolamento externa.
- Propriedades de dissipação de calor do condutor.
- Propriedades de dissipação de calor do isolamento.
- Propriedades de disspação de calor da malha de isolamento externa.

Estes factores, por sua vez, dependem de condições ambientais como a temperatura ambiente; a amplitude máxima das variações térmicas no meio; a densidade das cablagens instaladas nas esteiras, canalizações e afins; material isolador; e circulação de ar.

Em testes preliminares efectuados com uma extensão de 100m de cabo CAT5e UTP [Wailing 2001], submetendo em série cada condutor a uma corrente de 0,750A e considerando um tempo de estabilização de 24 h a uma temperatura ambiente de 23 graus Celsius (ambiente controlado), foram registadas temperaturas entre 40 e 49 graus Celsius por condutor. Estes números são excessivos e limitam de forma determinante a capacidade de transporte de energia dos cabos UTP CAT5.

Calcula-se que, para uma tensão de 51 V DC, com a injecção PoEP a ser feita apenas através de 2 pares (dos 4 existentes), cada par será "atravessado" por uma corrente de 0,7A ou 1A, caso se queira atingir os 30W ou os 40W previstos para o 802.3poep. Para 4 pares, os requerimentos baixam para 0,35A e 0,44A por par, respectivamente. Sendo a temperatura a principal limitação, considera-se que o PoEP deverá fornecer, no máximo, cerca de 36W em operação com 4 pares (caso se respeite o limite de 394mA imposto pela FCC por par de condutores AWG24, quando todos os 4 pares do cabo UTP CAT5 estão em uso). Contudo, mesmo esta estimativa é minada pela questão da temperatura que constitui, sem dúvida, o principal factor limitador na equação do PoEP.

Esquema de alimentação para 2 e 4 pares

Exemplificando para o caso da alimentação por 2 pares vs. 4 pares, e ignorando a degradação das condições RF do cabo (críticas para transmissão em *Gigabit Ethernet*) e as variações de impedância, considere-se para efeitos de prova um factor de dissipação de energia idêntico para 2 e 4 pares.

Assim, e considerando o cálculo da variação de temperatura do seguinte modo:

$$tr = Ploss.\Theta r$$

(variação de temperatura = perdas no condutor por dissipação térmica x resistência térmica do material)

Para 4 pares obtém-se:

$$tr_4p = Ploss_4p.\Theta r_4p = 0{,}394A^2.12{,}5\Omega.2.\Theta r_4p = 3{,}88W.\Theta r_4p$$

As perdas de energia por dissipação de calor num condutor atravessado por uma corrente DC, de acordo com a Lei de Joule, calculam-se pela fórmula $P = I^2.R$, daí $Ploss_4p = 0{,}394A^2.12{,}5\Omega.2$, onde o valor de 0,394 A é calculado com base na limitação imposta pela ISO/EIA/FCC de 0,197A por condutor para um cabo com 8 condutores AWG24, ou 4 pares, em uso – o que dá 0,394A para cada um dos 2 pares – e os 12,5 Ω são a resistência DC total da malha fechada (*loop*) formada numa configuração de 2 pares, para um canal de 100m.

Um "canal" dessa extensão pode ser formado pelos seguintes itens, cada um contribuindo para a resistência DC total:

- um cabo UTP Cat 5 de 90m a 60 graus Celsius
  (9,79 Ω por condutor, 19,59 Ω por malha segundo a norma ISO/TIA/EIA-568-B.2)
- 4 conectores (2 x 4 x 0,3 Ω cada, segundo a norma ANSI/TIA/EIA-568-B.2 , o que totaliza 2,4 Ω)
- *patch cable* ScTP AWG26 de 10m (2,8 Ω, segundo a norma TIA/EIA/IS-729)

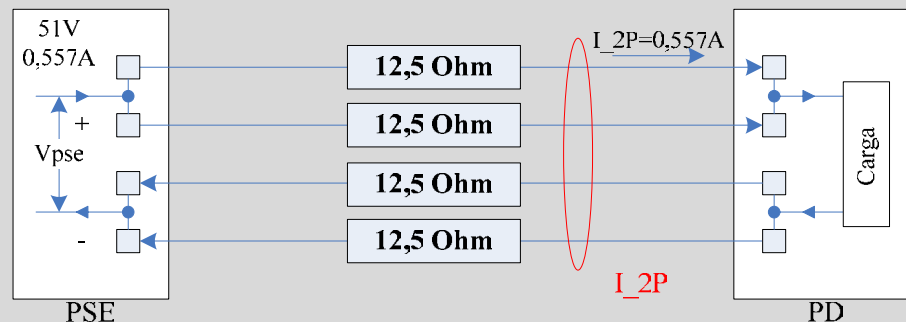
Estas três parcelas resultam num valor total de 24,79 Ω. Dividindo por 2 (pois estamos a lidar com malhas fechadas formadas por 1 par em cada sentido), obtém-se o valor final de 12,4 Ω. Contudo, em atenção à Secção 6.4.7 da norma ISO/IEC 11801:2002(E), está definido que para as categorias D, E e F (cablagem balanceada de cobre funcionando até 100, 250 e 650MHz e correspondendo aos requisitos de cablagem da categoria 5/5e, 6/6 *augmented* e 7, respectivamente) a resistência DC máxima por *loop* não deve nunca exceder 25 Ω. Assim sendo, consideramos um ajuste para 12,5 Ω (25/2), por respeito à norma, que nos permitirá trabalhar com cálculos em condições máximas.

Prosseguindo,

$$tr_4p = 3{,}88W.\Theta r_4p$$

Para 2 condutores, $tr_2p = Ploss_2p.\Theta r_2p = I^2.12{,}5\Omega.\Theta r_2p$

Logo, $tr_2p = I^2.12{,}5\Omega.\Theta r_2p$

Assumindo que *tr_2p=tr_4p* (dissipação térmica é idêntica para 2 e 4 pares):

$$I^2.12{,}5\Omega.\Theta r_2p = 3{,}88W.\Theta r_4p$$

$$I^2 = \frac{3{,}88W.\Theta r_4p}{12{,}5\Omega.\Theta r_2p} = \frac{0{,}3104.\Theta r_4p}{\Theta r_2p}$$

$$I = \sqrt{\frac{0{,}3104.\Theta r_4p}{\Theta r_2p}} = 0{,}557.\sqrt{\frac{\Theta r_4p}{\Theta r_2p}}$$

Se, e apenas se, $\Theta r_2p = \Theta r_4p$, então $I_2P = 0{,}557A$, pelo que a potência de alimentação para 2 pares seria de 24,5W (mas com uma corrente por par que excede as recomendações). Isto é facilmente determinado do seguinte modo:

Para cada malha, a energia perdida nos condutores é calculada por:

$$Ploss = 0{,}557A^2.51V = 3{,}88W$$

Logo a energia disponível na carga será igual à fornecida pela fonte, excepto perdas nos condutores:

$$Pforneccida = (51V.0{,}557A) - 3{,}88W = 24{,}5W$$

Contudo, o mais provável será que $\Theta r_2p > \Theta r_4p$ (devido a factores como a menor superfície total de dissipação térmica, devido ao menor número de condutores), o que aplicado à fórmula determina que:

$I_2P < 0{,}557A$ e portanto a potência de alimentação terá de ser menor que 24,5W

Para o caso em que existam uma ou mais linhas de *backup* directamente conectadas ao exterior é possível utilizar um adaptador VoIP FXS/FXO (incorporando as funções de *Analog Terminal Adaptor* e *gateway* PSTN), como o *Sipura* SPA-3000 (figura B.5).

Figura B.6 – Adaptador VoIP FXO/FXS

Recorrendo ao *Sipura* SPA-3000 é por exemplo possível o estabelecimento de uma configuração como a que se apresenta na Figura B.6. Nesta configuração, os terminais clássicos conectados ao adaptador (via conector FXS) operam com recurso à infraestrutura VoIP, apenas comutando para a linha tradicional (FXO) de *backup* em caso da indisponibilidade da rede IP.

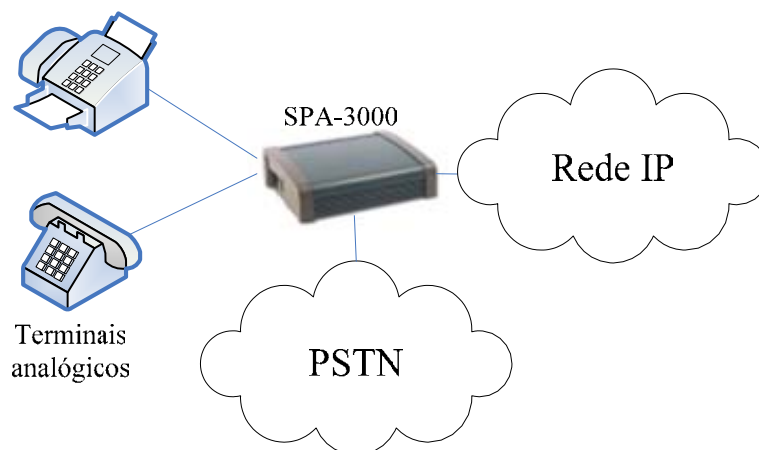



Figura B.7 – Configuração com Suporte para *Failover* Recorrendo ao Adaptador SPA-3000

B.4 Benefícios da Convergência Para VoIP

A estratégia IC^3 de convergência de serviços de comunicação e computação num só sistema aposta claramente na introdução progressiva de uma nova geração de SoIP (*Services Over IP*), dos quais o VoIP será apenas a ponta do icebergue. O objectivo é, desde o início, obter reduções de TCO (a nível local e global), minimizando os custos de migração e manutenção.

A manutenção de uma estrutura separada para voz e dados acarreta custos que vão da simples cablagem e estrutura passiva à manutenção dos postos terminais, passando pelo PBX, cuja configuração e manutenção é feita muitas vezes recorrendo à subcontratação dos serviços de um especialista. A migração para uma estrutura de SoIP permite obter ganhos consideráveis em vários aspectos que podem depender de cada organização, pelo que o devido planeamento e análise de ROI deverá ser levado a cabo antes de enveredar por esse caminho. No respeitante ao VoIP, especificamente, os benefícios e custos mais pertinentes estão assinalados na Tabela B.4.

Hard Cost **(*Capital Expenditure/Capex* e equipamento)**
Redução dos requisitos em cablagem na expansão da infraestrutura
Menor custo/tempo dispendido para adicionar/modificar/alterar a configuração da infraestrutura de terminais telefónicos, com menos recursos dispendidos no processo
Redução do investimento associado às várias fases do ciclo de vida do PBX tradicional (terminais, cartas de expansão, licenças, etc)
Eliminação dos custos de manutenção do PBX tradicional e sistemas associados
Redução no custo das chamadas, graças ao uso de recursos *in-band* (por exemplo, o uso de uma VPN permite que as chamadas entre filiais de uma empresa se façam internamente)
Adições/mudanças/alterações à estrutura de voz são simples, rápidas e eficientes.

Soft Cost **(produtividade/satisfação pessoal)**
Alocação de novos postos de trabalho com extensões IP é fácil de estabelecer e gerir, de acordo com as necessidades do negócio
Maior produtividade, graças ao uso da própria tecnologia para aumentar o *empowerment/*auto-suficiência dos utilizadores no desempenho das suas funções
Mobilidade acrescida, com a possibilidade de reprodução das condições do ambiente de trabalho de onde quer que estejam ligados à rede
Racionalização de recursos, com o fim da necessidade de equipas distintas para suporte da infraestrutura de dados e voz

Tabela B.4 – Factores Custo/Benefício do VoIP (adaptado de [Nóbrega 2005])

Referências

[ACPI4Linux] ACPI4Linux Project Homepage, http://acpi.sourceforge.net/documentation/sleep.html

[Aerospace] Aerospace Corporation, "How Forward Error-Correcting Codes Work", http://www.aero.org/publications/crosslink/winter2002/04_sidebar1.html

[Altiris] Altiris, http://www.altiris.com

[AMD] Advanced Micro Devices Inc, http://www.amd.com

[ARM] ARM Corporation, http://www.arm.com

[Asterisk] Asterisk project homepage, http://www.asterisk.org

[Berson 1992] Alex Berson, "Client/Server Architecture", McGraw-Hill Computer Science Series, pp. 11-12, 1992

[Blundon 1997] William Blundon, "Deciphering the NC World", JavaWorld, Mar/1997,(disponível em http://www.javaworld.com/javaworld/jw-03-1997/jw-03-blundon.html)

[Bogowitz 2005] Bob Bogowitz, Tracie Zenti, "Reducing costs with Intel Active Management Technology White Paper", http://www.intel.com/it/digital-enterprise/active-management-technology.pdf, Agosto de 2005

[Busch 2004] Doug Busch, Gregory Bryant, Bill Sayles, Tom Swinford, "The Digital Office: Cross-Platform Embedded IT for Manageability, Security and Conectivity", Technology Intel Magazine, Setembro de 2004

[Campbell 1998] Campbell, Richard, "Managing AFS: The Andrew File System", Prentice Hall, 1998

[Cisco] Cisco Systems, "Wireless Network Infrastructure", http://www.cisco.com/univercd/cc/td/doc/product/voice/c_ipphon/english/wip7920/7920ddg/wrlinfra.pdf

[Citrix] Citrix, Citrix Metaframe Homepage, http://www.citrix.com/

[Coda] Coda project homepage, http://www.coda.cs.cmu.edu/

[Coulouris 2001] George Couloris *et al*, "Distributed Systems – Concepts and Design (third edition)", Addison-Wesley Pearson, pp.324-325, 2001

[Cruz 2003] T. Cruz, P. Simões, "Enabling PreOS Desktop Management", Proceedings of IFIP/IEEE 8th International Symposium on Integrated Network Management (IM 2003),Colorado Springs, Março de 2003

[Cygwin] Cygwin project homepage, http://cygwin.com/

[Di Minico 2005] Chris Di Minico, "Call for Interest PoE-plus", Março de 2005

[DMTF 1998] DMTF, "Desktop Management Interface (DMI) v2.0", 1998

[DMTF 1999] DMTF, "Common Information Model (CIM) specification v2.2", 1999

[DMTF 1999a] DMTF, "Web-Based Enterprise Management", http://www.dmtf.org/standards/wbem/

[DMTF 2001] DMTF, "System Management BIOS Reference Specification v2.3.2", 2001

[DMTF 2001a] DMTF, "Alert Standard Format Specification v1.03", http://www.dmtf.org/standards/documents/ASF/DSP0114.pdf, Junho 2001

[DMTF 2002] DMTF, "Specification for the representation of CIM in XML, Version 2.1", http://www.dmtf.org/standards/documents/WBEM/DSP201.html, Maio de 2002

[DMTF 2002a] DMTF, "XML Document Type Definition v2.2", http://www.dmtf.org/standards/wbem/CIM_DTD_V22.dtd, 2002

[DMTF 2004] DMTF, "Specification for CIM Operations over HTTP, Version 1.2", http://www.dmtf.org/standards/published_documents/DSP0200.html, Dezembro

de 2004

[DMTF] DMTF, Distributed Management Task Force, http://www.dmtf.org

[Dolly] Dolly project homepage, http://www.cs.inf.ethz.ch/CoPs/patagonia/dolly.html

[Dolly+] Dolly+ project homepage, http://corvus.kek.jp/~manabe/pcf/dolly/

[Douglis 1993] Fred Douglis *et al.*,"Thwarting the power-hungry disk", USENIX Winter 1994 Technical Conference, 29 de Dezembro de 1993

[Feldman 1999] Jonathan Feldman, "Inside Intel's Wired for Management", http://www.networkcomputing.com/1020/1020ws1.html, Network Computing Magazine, Outubro de 1999

[Floyd 1995] Floyd, S., *et al.*,"Link-sharing and Resource Management Models for Packet Networks", IEEE/ACM Transactions on Networking, Vol. 3 No. 4, Agosto de 1995

[Foley 2005] Mary Jo Foley, "Microsoft Readies a 'Lean' Windows Client", Microsoft Watch, 12 de Maio de 2005

[Gartner 1997] Gartner Consulting, "TCO Analyst: A White Paper on GartnerGroup's Next Generation Total Cost of Ownership Methodology", 1997, pp. 19-20

[Gartner 1997a] Gartner Group, "TCO: New Technologies, New Benchmarks," Gartner Group's *Managing Distributed Computing Research Note* TCO-252, 5 Dezembro de 1997

[Gatekeeper] GNU Gatekeeper project homepage, http://www.gnugk.org

[Gemmell] J. Gemmel *et al.*, "FCast Multicast File Distribution", IEEE Network, Janeiro de 2000

[Gnomemeeting] Gnomemeeting project homepage, http://www.gnomemeeting.org

[Halfhill 1998] Tom Halfhill, "Crash-Proof Computing: Here's why today's PC's are the most crash-prone computers ever built – and how you can make yours more reliable", Byte Magazine, Abril de 1998

[Heartbeat] Linux High-Availability Homepage, http://www.linux-ha.org/Heartbeat

[Heilbronner 1997] S. Heilbronner, R. Wies, "Managing PC Networks", IEEE Communications Magazine, Outubro de 1997

[HTB] HTB-Linux project homepage, http://luxik.cdi.cz/~devik/qos/htb/

[IBM 1981] IBM Corporation press release, "Personal Computer announced by IBM", 12 de Agosto de 1981

[IBM 1993] IBM Corporation, "IBM PS/2 E product information", IBM DC 00210-00, 1993

[IEEE 2001] IEEE, "IEEE Std 802.1X: Port-Based Network Access Control", 14 de Junho de 2001

[IEEE 2003] IEEE P802.3af DTE Power via MDI Task Force, 2003, http://grouper.ieee.org/groups/802/3/af/

[IEEE 2004] IEEE 802.11 Working group, "IEEE Std 802.11i, Part 11, Amendment 6: Medium Access Control (MAC) Security Enhancements", 24 de Junho de 2004

[Igel 2005] IGEL Press Release, "IGEL Technology Demonstrates "thin" Voice over IP at CeBIT", http://www.igel.com/templates/lmcontent/powerslave,id,714,nodeid,170,_language,en.html

[iLabs 2002] iLabs Wireless Security Team, "What's wrong with WEP?", Networkworld, 9 de Setembro de 2002

[INRIA] INRIA Planète-BCAST home, http://planete-bcast.inrialpes.fr/rubrique.php3?id_rubrique=1

[Intel 1997] Intel Corporation, "MicroATX Motherboard Interface Specification", 1997

[Intel 1998] Intel Corporation, "Extensible Firmware Interface", http://developer.intel.com/technology/efi/, 1998

[Intel 1999] Intel Corporation, "FlexATX Addendum v1.0 to the microATX Specification", 1999

[Intel 1999a] Intel Corporation, "Intel Wired for Management", http://www.intel.com/technology/computing/wfm.htm

[Intel 1999b] Intel Corporation, "Preboot Execution Environment Specification 2.1", Setembro de 1998

[Intel 2005] Intel Corp, "Intel Stable Image Platform Program", http://www.intel.com/infot/stableplatform

[Intel 2005a] Intel Corp., "Intel Platform Solutions: Desktop Platform Vision", http://www.intel.com/platforms/desktop/vision/

[Intel 2005b] Intel Corp., "Platform 2015", http://www.intel.com/technology/computing/archinnov/platform2015/

[Intel 2005c] Intel Corp, "Balanced Technology Extended (BTX) Chassis Design Guidelines, Revision 1.0", Novembro de 2005

[Intel/Microsoft 1998] Intel Corp, Microsoft Corp, "PC Design Guide Website", http://www.microsoft.com/whdc/system/platform/pcdesign/desguide/pcguides.mspx, 1998

[Intel] Intel LANDesk Homepage, http://www.intel.com/network/products/landesk/

[Kphone] Kphone project homepage, http://sourceforge.net/projects/kphone

[Leach 1996] Paul Leach, Dan Perry, "CIFS: A Common Internet File System", Microsoft Internet Developer, Novembro de 1996

[Lidinsky 1999] W. Lidinsky, "IEEE Standard P802.1D Information tecnology - Telecommunications and information exchange between systems - Common specifications - Part 3: Media Access Control (MAC) Bridges", 14 de Março de 1999

[LINBIT] LINBIT IT DRDB, http://www.linbit.com/en/drbd/drbd/

[LinITX] LinITX forum, http://www.linitx.org/forum/viewtopic.php?t=1727&sid =f9e6744a8d16105debd25d269078a782

[Lorch 1995] Jacob Lorch, "A Complete Picture of the Energy Consumption of a Portable Computer", *Masters Thesis*, Computer Science, University of California at Berkeley, Dezembro de 1995

[LTSP] Linux Terminal Server Project, www.ltsp.org

[Lustre] Lustre Project Homepage, http://www.lustre.org/

[Mahesri 2004] Aqeel Mahesri *et al.* "Who Eats the Energy ? Power Consumption on a Modern Laptop", project report CS497YYZ, centre for reliable and high-performance computing, University of Illinois at Urbana.-Champaign, 2004

[Microsoft 1988] Microsoft Corp, "Microsoft Networks/SMB File Sharing Protocol Extensions,Version 2.0, Document Version 3.3", Novembro de 1988

[Microsoft 1996] Microsoft Corp, "DCOM Technical Overview", http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndcom/html/msdn_dcomtec.asp, Novembro de 1996

[Microsoft 1997] Microsoft Corporation, "Network PC System Design Guidelines v1.0b", 1997 (disponível em http://www.eu.microsoft.com/hwdev/archive/netpc.asp)

[Microsoft 1999] Microsoft Corporation, "Terminal Services: providing the benefits of remote application execution", White Paper, http://www.microsoft.com/windows2000/server/evaluation/business/terminal.asp , Dezembro de 1999

[Microsoft 1999a] Microsoft Corp., "Remote Operating System Installation Overview White Paper", http://www.microsoft.com/windows2000/techinfo/howitworks/management/remoteover.asp, Outubro de 1999

[Microsoft 2004] Microsoft Corp, "Software Update Services Overview White Paper", http://www.microsoft.com/windowsserversystem/updateservices/techinfo/previous/susoverview.mspx, Janeiro de 2004

[Microsoft 2004a] Microsoft Corp, "Windows Update Services 2.0 Discussion", http://www.microsoft.com/technet/community/chats/trans/security/sec0407.mspx , Abril de 2004

[Microsoft 2004b] Microsoft Corporation, "Microsoft Windows XP Embedded EWF Overview", 21 de Setembro de 2004, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xpehelp/html/xeconenhancedwritefilter.asp

[Microsoft 2005] Microsoft Corp, "WMI Reference Guide", http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wmisdk/wmi/wmi_reference.asp

[Microsoft 2005a] Microsoft Corp, "VBScript Primer", http://www.microsoft.com/technet/scriptcenter/guide/sas_vbs_overview.mspx

[Microsoft 2006] Microsoft Corporation, Windows Embedded Homepage, http://www.microsoft.com/windows/embedded/default.mspx

[Microsoft] Microsoft Systems Management Server (SMS) Homepage, http://www.microsoft.com/smserver/

[MIPS] MIPS Technologies Inc, http://www.mips.com

[Moore 1965] Gordon Moore, "Cramming more components onto Integrated Circuits", Electronics Magazine, vol. 38, num 8, Abril de 1965

[Myers 2005] Mike Myers, "Longhorn OS Deployment & Manufacturing Features White Paper", WinHEC 2005, Julho de 2005

[National] National Semiconductor Corporation, http://www.national.com

[NBD] Linux Network Block Device, http://nbd.sourceforge.net

[Nóbrega 2005] João Nóbrega, "A voz num "tom" mais barato", Bit Magazine, Setembro de 2005

[Nomachine] Nomachine, http://www.nomachine.com

[OnionNetworks] Onion Networks Inc. Swarmcast, http://swarmcast.net/

[OpenH323] OpenH323 project page, http://www.openh323.org

[OpenLDAP] OpenLDAP project page, http://www.openldap.org

[OpenSSH] OpenSSH project homepage, http://www.openssh.com

[OpenSSL] OpenSSL project homepage, http://www.openssl.org/

[Oracle 2001] Oracle Corp., "50 Defining Moments: The innovations, choices, and visions that made Oracle an enterprise technology powerhouse.", http://www.oracle.com/technology/oramag/oracle/01-nov/o61cover_50define.html, Ponto 19, "The Internet Vision, 1995"

[Patagonia] Patagonia project homepage, http://www.cs.inf.ethz.ch/CoPs/patagonia/

[Plastina 1997] Dan Plastina, "Microsoft Zero Administration for Windows", http://www.usenix.org/publications/library/proceedings/lisa97/invited_talks/scarr/tsld024.htm, invited talk, Usenix LISA97

[Pountain 1998] Dick Pountain, "Thin-Client Computing: As the hype dies down there's a clear business case for thin clients", Byte Magazine, Abril de 1998, pp. 13

[QEMU] QEMU Open-source processor emulator, http://fabrice.bellard.free.fr/qemu

[Rdesktop] Rdesktop project homepage, http://www.rdesktop.org

[Redhat 2002] RedHat Cluster Manager Overview, http://www.redhat.com/docs/manuals/enterprise/RHEL-3-Manual/cluster-suite/ch-rhcm-overview.html

[RedHat 2004] RedHat Global Filesystem, http://www.redhat.com/software/rha/gfs/

[Reed 1960] I. S. Reed *et al*., "Polynomial Codes Over Certain Finite Fields", SIAM Journal of Applied Mathematics, vol 8, 1960

[RFC 1157] RFC 1157, "Simple Network Management Protocol", http://www.faqs.org/rfcs/rfc1157.html

[RFC 1198] RFC 1198, "For Your Information on the X Window System", http://www.rfc-archive.org/getrfc.php?rfc=1198, Janeiro de 1991

[RFC 1235] IETF Network Working Group, "The Coherent File Distribution Protocol", Junho de 1991

[RFC 1350] IETF Network Working Group, "The TFTP Protocolo (Revision 2)", Julho de 1992

[RFC 1661] IETF Network Working Group, "RFC1661: The Point-to-Point Protocol (PPP)", Julho de 1994

[RFC 1813] IETF Network Working Group, "RFC1813: NFS Version 3 Protocol Specification", Junho de 1995

[RFC 1904] RFC 1904, "NFS: Network File System Protocol", http://www.ietf.org/rfc/rfc1094.txt

[RFC 2090] IETF Network Working Group, "TFTP Multicast Option", Fevereiro de 1997

[RFC 2251] RFC 2251, "Lightweight Directory Access Protocol v3", http://www.ietf.org/rfc/rfc2251.txt, Dezembro de 1997

[RFC 2475] IETF Network Working Group, "An Architecture for Differentiated Services", Dezembro de 1998

[RFC 2865] IETF Network Working Group, "RFC2865: Remote Authentication Dial In User Service (RADIUS)", Junho de 2000

[RFC 3261] IETF Network Working Group, "RFC3261: SIP: Session Initiation Protocol", Junho de 2002

[RFC 3450] IETF Network Working Group, "Asynchronous Layered Coding (ALC) Protocol Instantiation", Dezembro de 2002

[RFC 3530] IETF Network Working Group, "RFC3530: Network File System (NFS) Version 4", Abril de 2003

[RFC 3748] IETF Network Working Group, "RFC3748: Extensible Authentication Protocol (EAP)", Junho de 2004

[RFC 3926] IETF Network Working Group, "FLUTE - File Delivery over Unidirectional Transport", Outubro de 2004

[RFC 4301] IETF Network Working Group, "RFC4301: Security Architecture for the Internet Protocol", Dezembro de 2005

[Richardson 1998] Tristan Richardson, Kenneth Wood, "The RFB Protocol, Document Version 3.3", Julho de 1998

[Robinson 2002] Hawke Robinson, "Malware FAQ: Microsoft PPTP VPN", SANS Institute, 2002

[Rosensweig 1998] Phil Rosensweig *et al.*, "The Java Reliable Multicast Service: A Reliable Multicast Library", Sun Microsystems SMLI TR-98-68, Setembro de 1998,

[Rsync] Rsync project homepage, http://samba.anu.edu.au/rsync/

[Satyanarayanan 1989] M. Satyanarayanan, "Coda: A Highly Available File System for a Distributed Workstation Environment*",* Proceedings of the Second IEEE Workshop on Workstation Operating Systems, Setembro de 1989

[Schleifer 1996] Robert Schleifer, James Gettys, "X Window System: Core and Extension Protocols, X version 11 releases 6 and 6.1", Digital Press 1996

[Schneier 1999] Bruce Schneier *et al* "Cryptanalysis of Microsoft's PPTP Authentication Extensions (MS-CHAPv2)", 19 de Outubro de 1999

[Scribus] Scribus Open-Source Desktop Publishing project homepage, http://www.scribus.net

[Shapiro 2001] Jonathan S. Shapiro, "Capabilities", Block Lecture, Distinguished OS Lectures Series, University of Karlsruhe – lecture notes by Felix Hupfeld, 2001

[Sharpe 2002] Richard Sharpe, "Just what is SMB?" , 8 de Outubro de 2002, http://samba.anu.edu.au/cifs/docs/what-is-smb.html

[Steeleye] Steeleye Technology Lifekeeper, http://www.steeleye.com/

[Sultan 2005] Philippe Sultan, “SIP peers external authentication in Asterisk / OpenPBX“, 2005, http://www-rocq.inria.fr/who/Philippe.Sultan/Asterisk/asterisk_sip_external_authentication.html

[Sun 1996] Sun Microsystems, "Introducing Javastation:Sun's Enterprise Desktop Alternative to Transform Network Computing", http://www.sun.com/smi/Press/sunflash/1996-10/sunflash.961029.3088.html, Outubro de 1996

[Sun 2004] Sun Microsystems, “SunRay overview white paper”, http://www.sun.com/sunray/techinfo/New_SR_WP_12_04.pdf, Dezembro 2004

[UDPCast] UDPCast Project, http://www.udpcast.linux.lu

[UEFI] Unified Extensible Firmware Interface (UEFI) Forum, http://www.uefi.org

[v9fs] v9fs project homepage, http://swik.net/v9fs

[VIA 2001] VIA Technologies, “ITX Mainboard Specification White Paper”, 2001

[VIA 2001a] VIA Technologies, “Information PC reference design”, 2001

[VIA 2002] VIA Technologies, “Mini-ITX Mainboard Specification White Paper”, 2002

[VIA 2004] VIA Technologies, “VIA EPIA-N Motherboard”, http://www.viaembedded.com/product/epia_N_spec.jsp?motherboardId=221

[VIA 2004a] VIA Technologies, “VIA Nano-ITX Reference design history”, http://www.viaembedded.com/product/reference_design_story.jsp?motherboardId=182

[VIA 2004b] Via Technologies, “EPIA MII-Series Mini-ITX Mainboard Operating Guide, Version 1.1”, 18 de Novembro de 2004

[VNC] RealVNC project homepage, http://www.realvnc.com/

[W3C 2004] World Wide Web Consortium, “Extensible Markup Language (XML) version 1.1”, http://www.w3.org/TR/xml11/, Abril de 2004

[Wailing 2001] J. H. Wiling, “Current carrying capacity of data grade cables”, Beaconsfield, 24 de Maio de 2001

[Wallingford 2005] Ted Wallingford, “Switching to VoIP”, O’Reilly Media, Junho de 2005

[Weller 1999] L. Weller, “Reducing TCO with Intel WFM and Microsoft ZAW initiatives”,Intel Developer Update Magazine, Num. 17, Fev/1999

[Wi-Fi 2003] Wi-Fi Alliance, “Wi-Fi protected access: Strong, standards-based, interoperable security for today’s Wi-Fi networks”, 29 de Abril de 2003

[Wright 2004] Joshua Wright, asleap project homepage, 2004, http://asleap.sourceforge.net

[WYSE 2001] WYSE Technology, "In search of the Ideal Desktop: The Wyse thin-client Network Computing Vision", Maio de 2001

Fotografias: lista de fontes

As diversas fotografias incluídas nesta Dissertação são da autoria do candidato, com ressalva das seguintes excepções:

- Secção 2.2, Figura 2.1. Fonte: *IBM Corporation*.
- Secção 2.2, caixa da página 7. Fonte: *http://www.engcyclopedia.de.*
- Secção 2.3, Figura 2.3. Fonte: *Scientific American special edition*, "*The computer of the 21st Century*", Setembro de 1991.
- Secção 2.4.1, caixa da página 11. Fonte: *IBM Corporation*
- Secção 2.4.2, Figura 2.5 Fonte: *Byte Magazine, Sun Microsystems, Science & Vie Micro.*
- Secção 3.3.1, caixa da página 33. Fontes: *Intel Corporation*.
- Secção 4.2.3, caixa da página 47. Fontes: *Pretec Corporation*.
- Anexo B.3, Figura B.6. Fontes: *Sippura Incorporated*.